Machine Learning for Streamflow Prediction

by

Martin Gauch

A thesis
presented to the University of Waterloo
in fulfillment of the
thesis requirement for the degree of
Master of Mathematics
in
Computer Science

Waterloo, Ontario, Canada, 2020

I hereby declare that I am the sole author of this thesis. This is a true copy of the thesis, including any required final revisions, as accepted by my examiners.

I understand that my thesis may be made electronically available to the public.

Abstract

Accurate prediction of streamflow—the amount of water flowing past a stream section at a given time—is a long-standing challenge in hydrology. Not only do researchers strive to understand the natural processes at play, the predictions are also vital for management of floods, irrigation control, or hydro-electric power generation. Traditional, physically-based models explicitly simulate the processes that drive streamflow, but their predictions are often inaccurate, especially when predicting multiple watersheds with one model.

In this thesis, we study applications of machine learning to streamflow prediction: We present two case studies where data-driven models outperform physically-based models. Although more accurate, these data-driven techniques lack interpretability compared to physically-based models. Hence, we further explore first steps towards combining physically-based and data-driven approaches into a single model that preserves each component's advantages. Lastly, we quantify the effects of limited training data on the quality of data-driven predictions. We show that models benefit from additional data not only in terms of longer time periods, but also in terms of additional basins. This is a promising result towards transferring trained models to regions with limited or no training data.

As all of the above research directions hinge on the access to geospatial datasets, we precede their examination with the development of the Cuizinart, a cloud-based platform to disseminate and subset large environmental datasets.

Acknowledgements

I would like to thank my advisor Professor Jimmy Lin for his support and mentorship.

Further, I would like to thank everyone who helped making this thesis possible, especially Juliane Mai from the Department of Civil and Environmental Engineering. Lauren Fry and Emily Bradley from the U.S. Army Corps of Engineers and Shervan Gharari from the University of Saskatchewan shared their physically-based models' predictions; Nicolas Gasset and Vincent Fortin from Environment and Climate Change Canada provided the meteorological dataset for one of our case studies.

I would also like to thank Professor Frank Tompa and Professor Ihab Ilyas, who agreed to serve as readers of this thesis.

Lastly, I would like to thank the Global Water Futures project, the Integrated Modeling Program for Canada, and the Canada First Research Excellence Fund for their financial support, as well as Compute Ontario and Compute Canada for providing the computational resources that enabled our research.

Table of Contents

List of Tables viii

List of Figures ix

1 Introduction 1

1.1 Contributions 2

1.2 Thesis Organization 3

2 Background and Related Work 4

2.1 Streamflow 4

2.2 Streamflow Prediction 5

2.2.1 Nash–Sutcliffe-Efficiency 5

2.3 Physically-Based Streamflow Prediction 6

2.4 Data-Driven Streamflow Prediction 9

2.4.1 Tree-Based Models 9

2.4.2 Long Short-Term Memory Cell-Based Models 10

3 The Cuizinart: Environmental Data Subsetting 12

3.1 Architecture 13

3.2 The Canadian Surface Prediction Archive 14

4 Data-Driven vs. Physically-Based Models for Streamflow Prediction 16
4.1 Data . . . 16
4.2 Models . . . 17
4.3 Results . . . 19
4.4 Discussion . . . 22

5 Hybrid Physically-Based and Data-Driven Streamflow Prediction 23
5.1 Data . . . 24
5.2 Models . . . 26
5.3 Results . . . 28
5.4 Discussion . . . 32

6 Effects of Limited Training Data 33
6.1 Data and Method . . . 34
6.1.1 Data . . . 34
6.1.2 Training and Evaluation Procedures . . . 34
6.2 Results . . . 36
6.2.1 Input Sequence Length . . . 36
6.2.2 Training Period Length and Number of Basins . . . 37
6.3 Discussion . . . 43
6.3.1 Input Sequence Length . . . 43
6.3.2 Training Period Length and Number of Basins . . . 43

7 Conclusion 45

References 47

APPENDICES **52**

A Training Procedures 52

A.1 LSTM-Based Models 52

A.2 XGBoost Models 52

B Attributes in the CAMELS Dataset 53

List of Tables

3.1 Description, spatial extent, size, and time range of the data products available in the Cuizinart 13

4.1 Meteorological forcing variables in the RDRS data product 17

4.2 Minimum, median, and maximum of the NSE distributions for the physically-based model VIC-GRU and the data-driven models (XGBoost, LSTM, and ConvLSTM) 19

5.1 Number of calibration and validation basins, divided into low-impact and most-downstream basins 25

5.2 Meteorological forcing variables from the WFDEI-GEM-CaPA data product 25

5.3 Static basin characteristics we use in the Great Lakes case study 26

5.4 Minimum, median, and maximum of the NSE distributions for LBRM, XGBoost, LSTM, and LBRM+LSTM on the Great Lakes basins 28

6.1 Maurer meteorological variables in the CAMELS dataset 34

6.2 Input sequence lengths for XGBoost and EA-LSTM that yield the best median NSE for each number of basins and training years 37

6.3 Minimum, median, and maximum NSE scores and average percentage of failed basins (NSE ≤ 0) on the test period for XGBoost and EA-LSTM . . 38

A.1 Final XGBoost hyperparameters for each input sequence length on the CAMELS dataset 53

B.1 CAMELS static basin attributes used in this study 54

List of Figures

2.1 Schematic visualization of the abc-model as a simple, physically-based streamflow prediction model 7
2.2 Schematic illustration of the Large Basin Runoff Model 8
2.3 Schematic architecture of LSTM and EA-LSTM cells 11

3.1 Screenshot of the Cuizinart web application 14
3.2 Schematic diagram of the Cuizinart architecture 15

4.1 Map of the Lake Erie sub-basins and illustration of gridded forcings 18
4.2 Cumulative NSE distributions for VIC-GRU, LSTM, ConvLSTM, XGBoost 20
4.3 Time series of actual streamflow and predictions for VIC-GRU and XGBoost at gauging stations 04207200, 04177000, and 04166500 in the test period . 21

5.1 Map of the Great Lakes gauging stations in our analysis, divided into low human impact and most-downstream, and calibration and validation basins 24
5.2 Cumulative NSE distributions for LBRM, XGBoost, LSTM, LBRM+LSTM 29
5.3 Time series of actual streamflow and predictions for LBRM and LBRM+LSTM at gauging stations 04085427, 04126970, and 04119000 in the test period . 31

6.1 Cumulative NSE distributions for XGBoost and EA-LSTM 39
6.2 Relation between number of training samples and median NSE for XGBoost and EA-LSTM 41
6.3 Heatmap visualization of the XGBoost and EA-LSTM models' NSE values for each basin 42

Chapter 1

Introduction

Streamflow prediction—predicting the amount of water that flows past a point in a stream per time—is an important yet unsolved task in hydrology. Accurate streamflow predictions are vital in preparation to floods, as they allow authorities to proactively direct help to where it will be needed the most. With climate change, such extreme weather events are becoming more frequent. Streamflow predictions, however, are not only important in the event of flooding: During periods of drought, accurate predictions help planning river traffic, and even in times of modest flows, streamflow predictions provide valuable information to engineers managing hydroelectric facilities.

For decades, hydrologists have attempted to design physically-based models that predict streamflow based on meteorological and geophysical input data. In these models, researchers explicitly simulate physical processes and laws that govern the formation of streamflow, such as evaporation, percolation, or snow accumulation. In many cases, however, these models result in rather inaccurate predictions—especially when hydrologists use a single model to predict streamflow at different spatial locations.

As in other physical sciences, recent data-driven techniques promise rapid advancement in our prediction capabilities. From this viewpoint of data science, we face a time-series regression problem with spatio-temporally distributed input and output data. In this thesis, we explore the potential of applying machine learning to streamflow prediction. We substantiate our claim about the superior quality of data-driven predictions in two case studies, where data-driven models outperform existing physically-based models. As algorithms in machine learning often rely on large training datasets—which are not available in all regions of the world—we evaluate the effects of limited amounts of training data on prediction quality. In longer-term efforts, we see great potential in a confluence of

physically-based and data-driven models, known as theory-guided machine learning. As a first step towards such hybrid modeling, we demonstrate how data-driven models can correct a physically-based model's errors. The resulting joint model is more accurate than both physically-based and data-driven models individually, yet it maintains a notion of hydrologic interpretability.

The development of state-of-the-art hydrologic models as we design and employ in this thesis, both data-driven and physically-based, greatly relies on researchers' access to a variety of environmental datasets. These datasets are often extremely large, although individual researchers commonly require only a small subset of the whole data for their specific purposes. This situation results in a waste of storage space, bandwidth, and labor, as researchers have to download the whole dataset and manually crop it to their region and time period of interest. To this end, we develop the Cuizinart as a web-based tool that facilitates the access to custom subsets of environmental datasets.

1.1 Contributions

To summarize, this thesis makes the following main contributions:

- We present the Cuizinart, a cloud-based platform that allows users to request spatial and temporal subsets of large environmental datasets through an interface similar to Google Maps.
- We hypothesized that data-driven streamflow prediction models can outperform existing physically-based models. Our results confirm that this is the case, as we design data-driven models that yield more accurate predictions than a physically-based model in a case study on the Lake Erie watershed.
- We then hypothesized that information from physically-based models can further improve a data-driven model's accuracy. We propose a hybrid architecture, where an LSTM model obtains physically-based states as additional input. Our results show that the joint model outperforms models of either individual paradigm in a case study on the Great Lakes watersheds.
- As the data-driven models in our experiments appear sensitive to limited amounts of training data, we finally aimed to quantify how different aspects of training set size affect the quality of predictions. We show that model accuracy depends not only on the training period length, but also on the number of basins that we use when training on a benchmark dataset of the continental United States.

1.2 Thesis Organization

Following this introduction, Chapter 2 establishes important concepts and related work in hydrology and machine learning. The subsequent chapters align with our contributions:

- Chapter 3 highlights our efforts in alleviating scientific data access and subsetting through the Cuizinart.
- Chapter 4 describes our research around purely data-driven models for streamflow prediction and their comparison to existing physically-based models.
- Chapter 5 reports on our efforts towards hybrid physically-based and data-driven streamflow prediction models.
- Chapter 6 studies the effects that limited training data have on data-driven streamflow prediction.

Lastly, Chapter 7 concludes with final remarks and gives an outlook towards possible directions of future research.

Chapter 2

Background and Related Work

2.1 Streamflow

Streamflow measures the amount of water flowing past a given point along a river or stream per time. It is therefore measured as volume per time, usually in $m^3 s^{-1}$. Hydrologists measure a river section's actual streamflow at stream gauging stations. In many countries, government agencies such as the United States Geological Survey (USGS) maintain these sites and publish daily or even real-time measurements online.[1]

Given a point along a stream (whether gauged or not), we can delineate the corresponding *basin*—also known as *catchment* or *watershed*—as the upstream area that contributes to this location's streamflow. The given point is called its *outlet*. It can be the point where a river flows into a lake or the ocean, but it can also be an arbitrary point along the stream. A basin's area is largely defined by the surrounding topography: Intuitively, one can think of it as the region where every drop of precipitation will eventually flow past the outlet, either after running along the topographic slopes into the upstream river or after percolating into the groundwater and draining into the stream from there. Hence, the further upstream along a river we define an outlet, the smaller the corresponding basin. In addition, rivers and lakes drain into each other. Consequently, basins contain nested *sub-basins* (or *sub-catchments* or *sub-watersheds*). If there exists a gauging station that captures a *hydrograph*—the time series of actual streamflow—at the outlet of a specific catchment, we call the catchment a *gauged basin*, else, an *ungauged basin*. Hydrologists know predictions in ungauged basins (PUB) to be particularly challenging, since there are no immediate training data available for the evaluated basins [18].

[1] https://waterwatch.usgs.gov/

2.2 Streamflow Prediction

To predict streamflow, hydrologists use two types of input data: meteorological time-series data such as precipitation, temperature, or wind speed, and static basin attributes such as slope, elevation, or land cover data. The former time-series data are also known as *forcings*, as we use them to run, or *force*, a model. We can distinguish three broad types of model architectures:

Lumped models make use of the fact that only precipitation within the boundaries of a basin can affect its outlet's streamflow. These models take as input basin-aggregated forcings and basin characteristics. One example for a lumped model in operational use is the *Large Basin Runoff Model* operated by the U.S. Army Corps of Engineers to monitor streamflow in the Great Lakes region [9].

Distributed models operate on the level of grid cells. These models take as input forcings and static attributes for cells on a spatial grid of $h \times w$ cells. An example for a distributed model is the *mesoscale Hydrologic Model* (mHM) [39].

Semi-distributed models conceptually fall between lumped and fully-distributed models. They do not operate on uniform grid cells, but rather on modeling units of arbitrary (usually polygonal) shapes. Within a modeling unit, we assume that the geophysical characteristics are uniform, which can make computations faster than in the fully-distributed case, where we process each cell individually—even if adjacent cells have identical properties. Examples for semi-distributed models are the *Variable Infiltration Capacity model* (VIC) and its variant VIC-GRU, which uses modeling units called *Grouped Response Units* [15, 26].

2.2.1 Nash–Sutcliffe-Efficiency

Hydrologists commonly measure the quality of a model's streamflow predictions using a metric called *Nash–Sutcliffe Efficiency coefficient*, or NSE. The NSE compares a time series of predicted streamflow $\hat{y}_t$ with the measured hydrograph of ground truth streamflow y_t for the time steps $t = 1, \dots, T$, and is defined as:

$$\text{NSE} = 1 - \frac{\sum_{t=1}^{T} (\hat{y}_t - y_t)^2}{\sum_{t=1}^{T} (y_t - \bar{y})^2} \tag{2.1}$$

where $\bar{y}$ denotes the mean measured streamflow. NSE values range from $-\infty$ to 1, with 1 being the optimal value. An NSE of zero corresponds to a model that constantly predicts the mean streamflow $\bar{y}$.

As the numerator in Equation (2.1) measures the squared error and the denominator measures the variance of streamflow observations, NSE is strongly correlated with the mean squared error (MSE), as Equation (2.2) shows [13]:

$$\text{NSE} = 1 - \frac{\text{MSE}}{\frac{1}{T}\sum_{t=1}^{T}(y_t - \bar{y})^2} \tag{2.2}$$

To use NSE as the loss function of a neural network or gradient-boosted regression tree, we extend its definition to a set B of basins:

$$\text{NSE}' = \frac{1}{|B|}\sum_{b \in B}\sum_{t=1}^{T^b}\frac{(\hat{y}_t^b - y_t^b)^2}{(\sigma^b + \epsilon)^2} \tag{2.3}$$

where T^b is the number of time steps for which we have data on basin b, $\hat{y}_t^b$ and y_t^b are predicted and actual streamflow at basin b and time t, and σ^b is the standard deviation of observed streamflow at basin b. The addition of $\epsilon > 0$ ensures numeric stability, as it prevents NSE' from exploding for basins with near-constant flows.

2.3 Physically-Based Streamflow Prediction

Traditionally, hydrologists have been modeling streamflow using *physically-based models*. In these models, researchers try to explicitly formulate physical processes that drive the incurrence of streamflow, such as evaporation, transpiration, and percolation of water.

Many physically-based models operate on the basis of so-called *states* (or *storages*) that work as "memory cells". Each state has an initial value that the model equations update depending on the input.

Figure 2.1 depicts a simple (educational) physically-based model, the *abc-model*, named after its three parameters [10]: the fraction a of precipitation that percolates into the groundwater storage, the fraction b of precipitation that evapotranspirates into the atmosphere, and the fraction c of the groundwater storage that drains into the stream at each

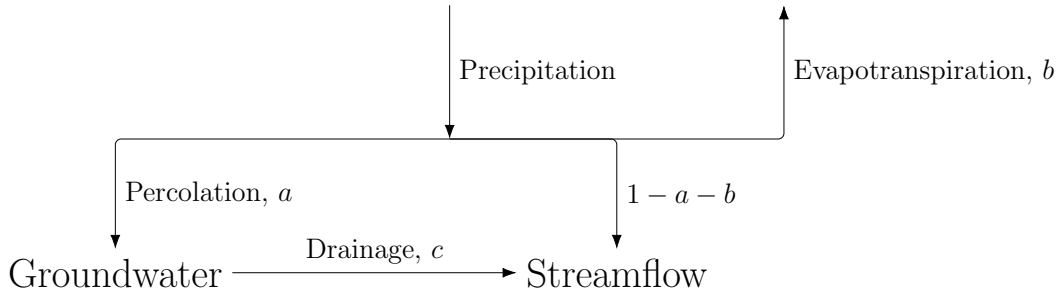


Figure 2.1: Schematic visualization of the abc-model as a simple, physically-based streamflow prediction model. At each time step, incoming precipitation either percolates into the groundwater storage (a), evapotranspirates into the atmosphere (b), or directly runs off into the stream ($1 - a - b$). A fraction of the groundwater storage (c) contributes to streamflow as drainage.

time step. The schematic model illustration in Figure 2.1 therefore translates into the following equations:

$$\begin{aligned} \hat{y}_t &= (1 - a - b)x_t + cG_{t-1} \\ G_t &= (1 - c)G_{t-1} + ax_t \end{aligned} \tag{2.4}$$

where x_t represents the precipitation, G_t refers to the groundwater storage, and $\hat{y}_t$ denotes the predicted streamflow at time t.

In this thesis, we will make use of two more sophisticated physically-based models, the lumped Large Basin Runoff Model (LBRM) [9] and the semi-distributed *Variable Infiltration Capacity model based on Grouped Response Units* (VIC-GRU) [15, 26]. The U.S. Army Corps of Engineers maintains LBRM and uses it operationally for streamflow predictions in the Great Lakes region. Figure 2.2 outlines the structure of LBRM.

To provide good predictions after training (hydrologists call this the *calibration* period), it is key that we adjust the initial model states to the beginning of the test period. For instance, an initial snow depth state of zero might not be suitable for a test period in winter. Hydrologists ensure good initial states by prepending a *warm-up* phase before the actual test (or *validation*) period. During warm-up, the model ingests input data and updates its states based on the input, however, the generated output is not considered in evaluation metrics.

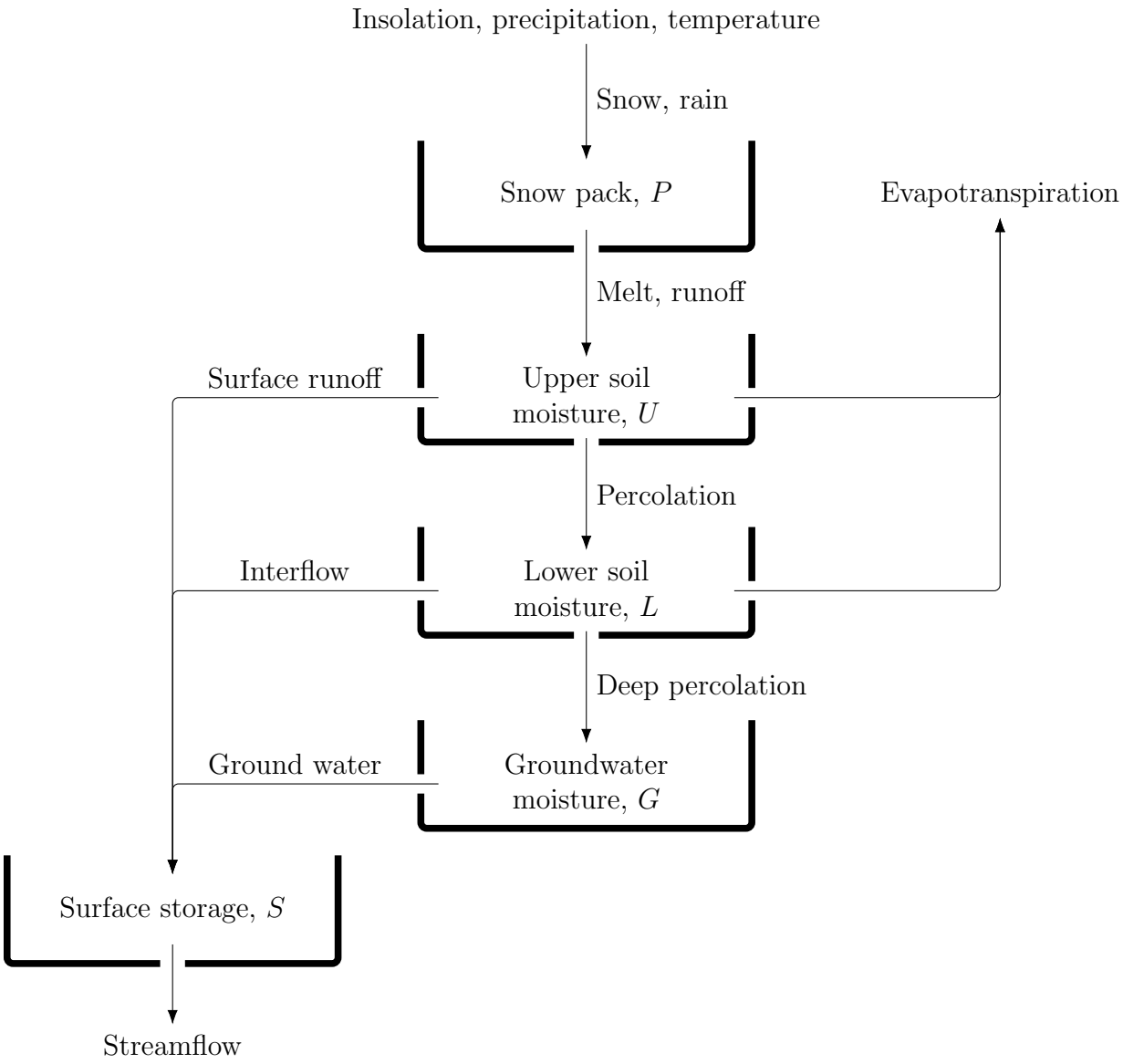


Figure 2.2: Schematic illustration of the Large Basin Runoff Model (LBRM). (Illustration derived from Croley [9])

2.4 Data-Driven Streamflow Prediction

Since the overwhelming success of machine learning in domains such as image recognition, machine translation, and recommender systems, recent years have shown increasing adoption of techniques based on machine learning in the physical sciences. Examples for this range from materials science, where researchers predict properties of material specimen or chemical components of entirely new materials [27], to high-energy particle physics, where researchers detect hypothesized particles through deep learning [5]. In many cases, these machine-learned approaches to long-standing problems result in models that outperform traditional solutions. In light of this situation, it seems natural to develop data-driven models for streamflow prediction—and in fact, applications of neural networks to this or related problems date back decades [4, 6, 14]. Until recently, however, such approaches only had limited success in a few, individual basins. Newer algorithms, such as Long Short-Term Memory networks, combined with today's computational power, make it worthwhile to revisit data-driven streamflow prediction. Indeed, recent studies have found machine learning models to provide accurate predictions even when trained on multiple basins at once—a task with which physically-based models commonly struggle [25].

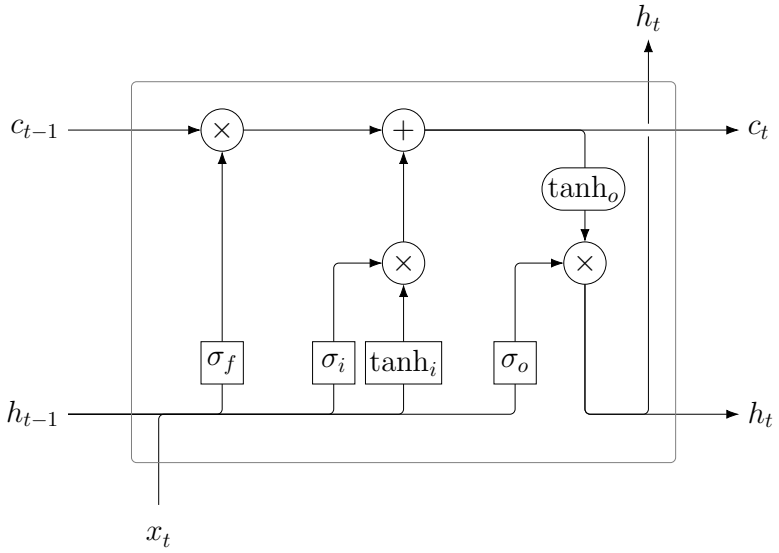
2.4.1 Tree-Based Models

Tree-based models have proven successful in a variety of time-series prediction tasks—although they are not explicitly designed for time-series input. For instance, practitioners have used XGBoost, a framework for *gradient-boosted regression trees* (GBRT) [7], in numerous Kaggle and KDD Cup challenges as part of the winning solutions [20, 21, 22, 28]. Regression tree algorithms learn a graphical tree structure, where every leaf node corresponds to a prediction and every inner node represents a condition on an input feature. A path from the root to a leaf consequently equates to a conjunction of features. Samples whose features let the conjunction evaluate to `True` receive the leaf value as their prediction. To avoid overfitting, regression trees commonly use regularization methods such as limits on the maximum tree depth. GBRT extend this algorithm, as they sequentially train multiple regression trees to predict the residual error of all previous trees' predictions. The overall prediction is therefore the sum of all trees' predictions. Further, GBRT typically employ regularization as part of the loss function they optimize during training.

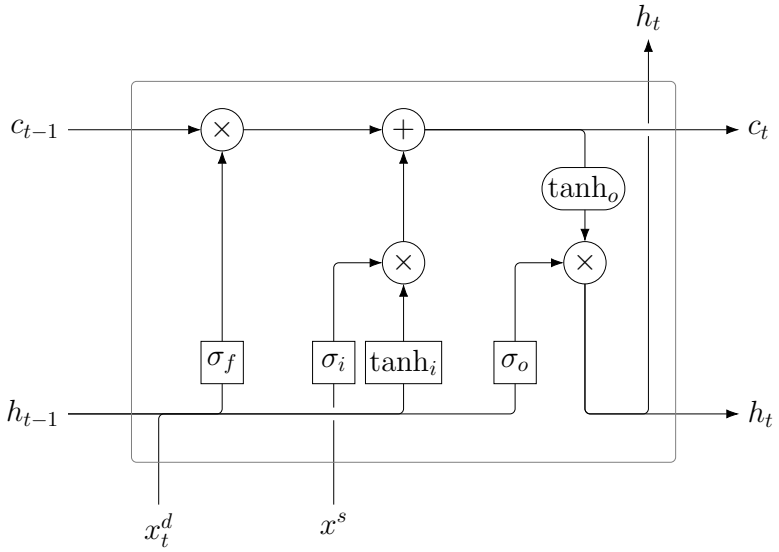
2.4.2 Long Short-Term Memory Cell-Based Models

From the perspective of data science, streamflow prediction is a time-series regression task. Hence, Long Short-Term Memory cell (LSTM)-based models are a natural choice for this problem [17]. Figure 2.3A shows the schematic architecture of an LSTM cell, with three major building blocks: the sigmoid-activated layer σ_f controls the *forget gate*, which deletes parts of the previous step's internal state information c_{t-1}; an *input gate* (σ_i, $\tanh_i$) controls how the input x_t updates the internal state; finally, an *output gate* (σ_o, $\tanh_o$) determines how internal state and new input affect the cell output h_t. In a recent study, Kratzert et al. [25] introduce an *Entity-Aware LSTM* (EA-LSTM) architecture as shown in Figure 2.3B, which they specifically design for streamflow prediction. Its cells take both time-series forcings x_t^d and static catchment attributes x^s as individual inputs, rather than requiring to concatenate the static information to each forcing time step. Their study on basins across the continental United States shows that LSTM-based models consistently outperform physically-based models.

An extension that allows us to use LSTMs on not just temporally, but also spatially distributed data are *convolutional LSTMs* (ConvLSTMs) [41]. While the principal idea and structure remain identical to standard LSTMs, ConvLSTMs take as input a time series of rectangular grids. The main difference to the standard LSTM in Figure 2.3A is that the cell applies convolution operations over the input x_t and previous step's output h_{t-1}. Subsequently, the data that flows through the network maintains its grid-shape, which results in gridded outputs.



(2.3A) LSTM cell.



(2.3B) EA-LSTM cell.

Figure 2.3: Schematic architecture of LSTM and EA-LSTM cells. The rectangles σ_* and $\tanh_*$ represent sigmoid- and tanh-activated neural layers; rounded shapes denote pointwise multiplication ($\times$), addition ($+$), and tanh operations. Unlike a vanilla LSTM, the EA-LSTM takes dynamic time-series forcings x_t^d and static basin attributes x^s as distinct input vectors. (Illustrations derived from Olah [37] and Kratzert et al. [25])

Chapter 3

The Cuizinart: Environmental Data Subsetting

Both traditional and data-driven hydrologic modeling largely depend on the processing of geospatial datasets—yet another example of data-intensive science emerging as the "fourth paradigm" of research [16]. Often, these datasets are scattered across numerous websites, use different formats, and contain larger temporal and spatial extents than an individual researcher requires. Ultimately, this results in a waste of time, bandwidth, and labor, as researchers have to individually download, normalize, and subset the datasets they need.

This chapter introduces our efforts towards facilitating these necessary steps of data gathering and preprocessing. We develop the Cuizinart (http://cuizinart.io/) as a web application with an interface similar to Google Maps, where researchers can request arbitrary subsets of commonly used environmental datasets. Figure 3.1 shows a screenshot of the Cuizinart web application. Users can choose from a list of data products and select their date range and variables of interest. Table 3.1 describes the available data products and their temporal and spatial extent. Upon selection of a data product, the map interface shows the product's geographic extent, and users can either draw their desired spatial subset on the map or upload it as a shapefile or GeoJSON specification.

When the user sends the request to the Cuizinart back end, it subsets the data product according to the specification and stores the results in NetCDF files.[1] When this processing completes, the user receives an email with a download link through the Globus large file transfer service.[2]

[1] https://www.unidata.ucar.edu/software/netcdf/
[2] https://www.globus.org/

Table 3.1: Description, spatial extent, size, and time range of the data products currently available in the Cuizinart.

Product	Explanation	Spatial extent	Size	Time range	
				from	to
`ERA5`	climate reanalysis	worldwide	615 GB	2002	2017
`pgw-wrf-wca`	global warming climate simulation	western Canada	1.6 TB	2000	2015
`ctl-wrf-wca`	retrospective climate simulation	western Canada	1.6 TB	2000	2015
`pgw-wrf-conus`	global warming climate simulation	continental U.S.	4.3 TB	2000	2013
`ctl-wrf-conus`	retrospective climate simulation	continental U.S.	4.4 TB	2000	2013
`wfdei-gem-capa`	meteorological forcings	$\sim$ North America	621 GB	1979	2017
`canrcm4-wfdei-gem-capa`	meteorological forcings	$\sim$ North America	16 TB	1951	2100
`MODIS-lst-mod-myd11_merged`	meteorological forcings	Great Lakes	65 GB	2002	2017

3.1 Architecture

The Cuizinart consists of two major components: a lightweight front end, implemented in JavaScript (React) and Python (Flask) docker containers, and a back end, implemented in Python. Figure 3.2 shows a schematic architecture diagram. The front end stores available product names and properties in a PostgreSQL database. Whenever a user loads the Cuizinart website, the Flask application reads the available product metadata from the database and sends them to the React application for display.

Once a user sends a request for a product, the Flask application forwards it via `scp` as a JSON file to the back end. We use `scp` rather than HTTP (which might seem like a more natural fit for a web service), since we host the back end on Compute Canada's Graham high-performance computing infrastructure, where HTTP access to server nodes is discouraged. The back end receives the request, schedules a job to subset the specified product, and, once completed, notifies the user via email that the requested data is ready to download through the Globus file transfer service.

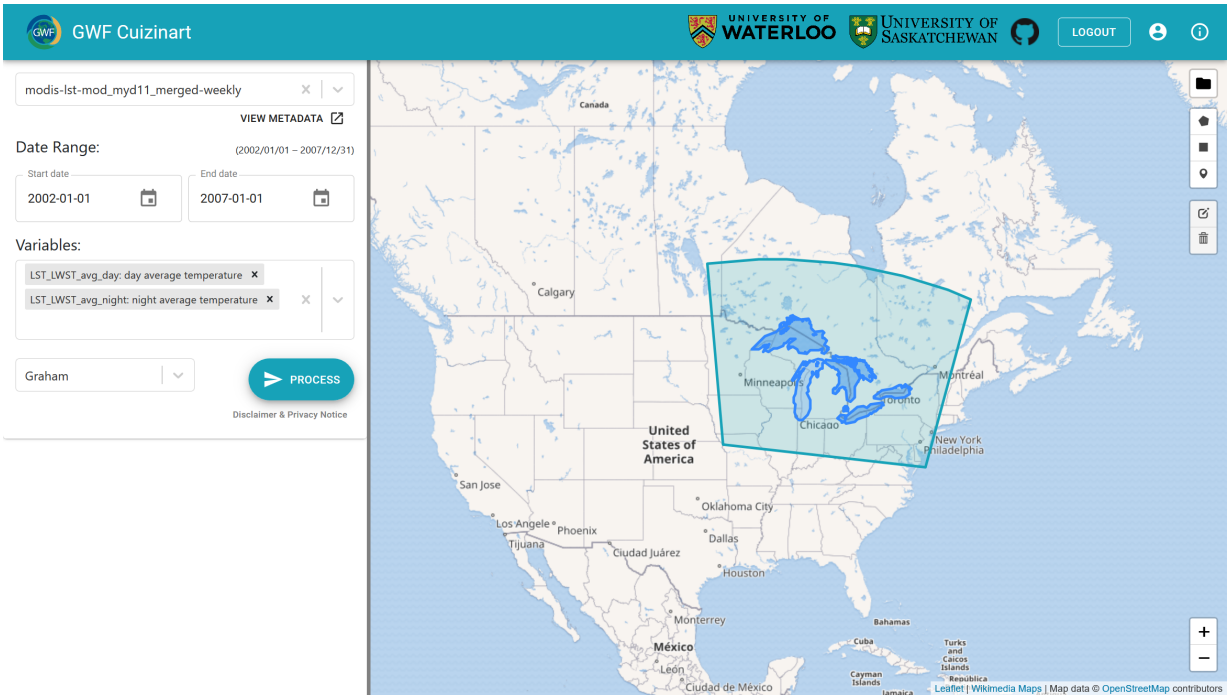


Figure 3.1: Screenshot of the Cuizinart web application. In this example, the user has selected a product that covers the Great Lakes region and uploaded a shapefile for the boundaries of the Great Lakes.

3.2 The Canadian Surface Prediction Archive

The Cuizinart's modular, dockerized architecture allows us to reuse its codebase in other applications. Most notably, we use the Cuizinart architecture in a separate project for the Canadian Surface Prediction Archive (CaSPAr) front end. CaSPAr has a similar scope as the Cuizinart, but it provides access to other types of environmental datasets—in this case, Environment and Climate Change Canada's numerical weather forecasts. In December 2019, CaSPAr held 318 TB of archived data, with 368 GB of newly generated predictions added every night.

Previously, the CaSPAr front end used proprietary map software provided and managed by esri. Consequently, each addition or update of a data product required error-prone manual work by esri support staff. With our new front end, a REST API allows automatic ingestion of additional products as well as incremental updates to these products. Each day, the back end receives Environment and Climate Change Canada's newly issued weather

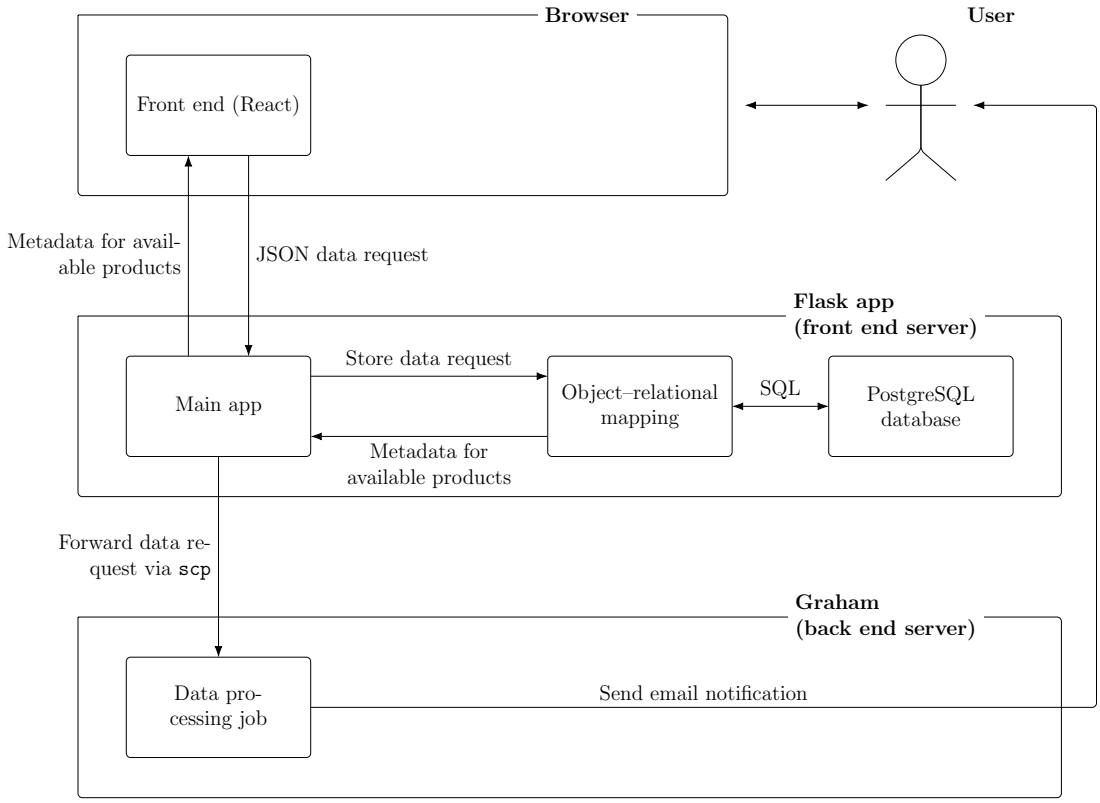


Figure 3.2: Schematic diagram of the Cuizinart architecture and control flow.

predictions as NetCDF files, analyzes their contents, and sends a POST request that describes the added data to the front end's REST API. The front end updates its metadata database and subsequently offers the updated products to end users.

Chapter 4

Data-Driven vs. Physically-Based Models for Streamflow Prediction

Given the wide range of existing streamflow prediction models, it is often unclear which model is best under which conditions. Studies that examine individual models use different basins, different time ranges, and different input datasets. The Great Lakes Intercomparison Project (GRIP) is the largest Canadian effort yet to overcome these problems and to evaluate the performance of different streamflow prediction models under identical conditions [29, 30]. While most of the evaluated models are physically-based, we contribute several data-driven models to the project. These data-driven models allow us to estimate the amount of information we can extract purely from the available data, and as such act as a benchmark to the physically-based approaches.

In the first phase, the intercomparison project focused on the Lake Erie watershed and used a small dataset of only five years of forcings.

4.1 Data

The meteorological forcing dataset we use in this case study is the Regional Deterministic Reforecast System (RDRS) product, provided by Environment and Climate Change Canada. It covers the Lake Erie region at a resolution of 15 km and provides information on temperature, precipitation, radiation, pressure, and wind (see Table 4.1) for the years 2010 to 2014. Figure 4.1A shows an example snapshot of the air temperature in the dataset.

Table 4.1: Meteorological forcing variables in the RDRS data product. Each variable covers the entire Lake Erie watershed at a resolution of around 15 km for the years 2010 to 2014 at an hourly resolution. The variables are available at the indicated vertical levels.

Variable	Explanation	Level	Unit
PR0	Quantity of precipitation	surface	m
TT	Air temperature	40 m	°C
FB	Downward solar flux	40 m	$\mathrm{W\,m^{-2}}$
FI	Surface incoming infrared flux	40 m	$\mathrm{W\,m^{-2}}$
P0	Surface pressure	surface	mbar
HU	Specific humidity	40 m	$\mathrm{kg\,kg^{-1}}$
UVC	Wind speed	40 m	kn

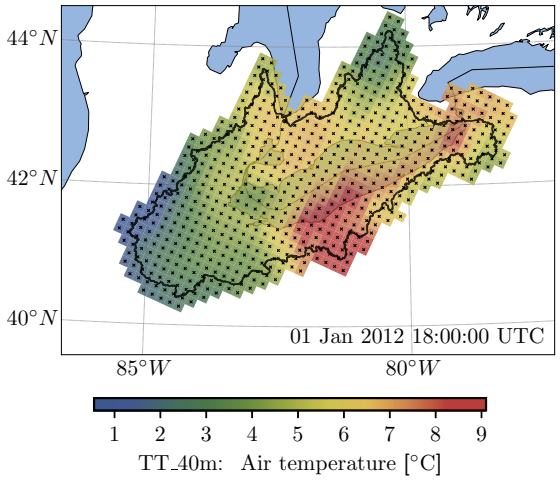
As streamflow ground truth, we use 46 gauged basins, which all eventually drain into Lake Erie. Figure 4.1B shows the basin outlines with their associated gauging stations. At the project outset, these basins were meant to be of low human impact, but over the course of the study it became clear that several of the basins are intensely human-managed and, for instance, located in highly urbanized areas.

4.2 Models

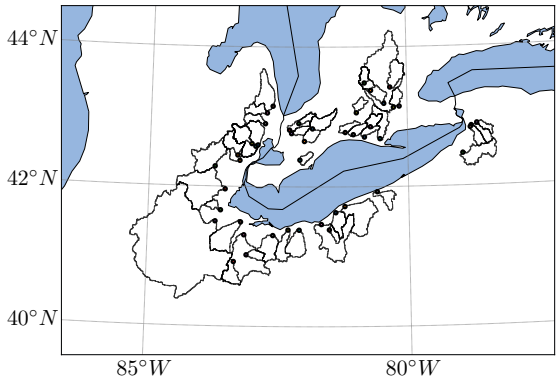
We compare VIC-GRU [15, 26], an existing semi-distributed physically-based model, with three data-driven architectures: the tree-based XGBoost, a standard LSTM, and a convolutional LSTM (ConvLSTM) network.

XGBoost. Since XGBoost does not directly consume time series, we flatten the data to fixed-history windows of eight days' forcings. Further, we aggregate the hourly temperature and precipitation to daily values (minimum, maximum, sum) in order to match the target streamflow data resolution. In preliminary experiments, the remaining forcing variables did not improve prediction quality; we therefore excluded them from the input. When we train one model to predict all basins combined, we additionally provide the model with one-hot-encoded station identifiers. To obtain suitable parameters, we perform a cross-validated random search.

LSTM. The standard LSTM has two layers of 128 hidden states. We feed the model the previous five days of hourly precipitation and temperature, as well as one-hot-encoded station and month identifiers. As loss function, we use $1 - \mathrm{NSE}$.



(4.1A) Gridded forcing data for the Lake Erie watershed (black outline). As an example, we show the temperature of Jan 1, 2012 18:00 UTC (colored tiles). Each tile is about $15 \times 15\,\mathrm{km}^2$ in size.



(4.1B) Geographical outlines of the 46 Lake Erie sub-basins in our analysis, each draining towards a gauging station (black dots).

ConvLSTM. A convolutional LSTM (ConvLSTM) network can better incorporate spatial input data, as its design allows it to ingest gridded data [41]. Our model consists of four convolutional LSTM layers followed by four convolutional but non-recurrent layers. To obtain predictions, we feed the history of the last eight days' precipitation, minimum and maximum temperature, combined with the one-hot-encoded month representation as a gridded matrix into the convolutional LSTM layers. We concatenate the last layer's last output with the station identifiers and pass the resulting tensor through the non-recurrent convolutions, using leaky rectified linear unit (ReLU) activation functions. Finally, these layers output a prediction for each grid cell. We select the cells that contain gauging stations and calculate the loss for each station as $1 - \mathrm{NSE}$.

As VIC-GRU and the ConvLSTM directly ingest grids of spatially-distributed data, we train them on all stations combined. For XGBoost and LSTM, we once train one model for each basin individually, and once one model on all basins' data combined.

Table 4.2: Minimum, median, and maximum of the NSE distributions for the physically-based model VIC-GRU and the data-driven models (XGBoost, LSTM, and ConvLSTM, either trained once for all stations or for each station individually). Best values are highlighted in bold.

	VIC-GRU	XGBoost		LSTM		ConvLSTM
Statistic	all stations	per-station	all stations	per-station	all stations	all stations
Min	−6.30	−0.21	**0.21**	−0.68	−0.32	−0.20
Median	0.33	**0.52**	0.49	0.12	0.18	0.43
Max	0.60	**0.67**	0.66	0.40	0.31	0.60

4.3 Results

Table 4.2 shows the minimum, median, and maximum prediction accuracy for each model in terms of NSE. The XGBoost model provides the best predictions. When we train one XGBoost model per basin, the median and maximum NSE results improve further, but the minimum NSE decreases. The LSTM and ConvLSTM show worse predictions than XGBoost. Likely, this is the result of sparse training data, combined with the high input dimensionality induced by the gridded input variables. Nevertheless, XGBoost and ConvLSTM outperform the physically-based VIC-GRU. XGBoost shows higher minimum, median, and maximum NSE values; the ConvLSTM reaches higher minimum and median NSEs and comparable maximum NSE values.

Figure 4.2 visualizes the NSE values across all 46 basins for the models trained on all stations combined in a cumulative distribution plot and denotes the corresponding area A under the distribution curve. Ideally, the curves would show a "⌟"-shape, with all basins' NSEs close to one. This would correspond to an area A under the curve of zero. For worse models, the distribution shifts leftward, which results in a larger area under the curve. Due to its poor lowest NSE values, VIC-GRU has the highest area under the curve ($A_{\text{VIC-GRU}} = 1.00$), although the bulk of its NSE values is better than the LSTM's ($A_{\text{LSTM}} = 0.89$). XGBoost has the lowest area ($A_{\text{XGBoost}} = 0.52$).

To further analyze the results, we focus on three basins: one where both XGBoost and VIC-GRU result in relatively accurate predictions, one where XGBoost outperforms VIC-GRU, and one where VIC-GRU outperforms XGBoost. Each row in Figure 4.3 displays the hydrograph and the models' predictions for one of these basins, as well as a map showing the basin's location.

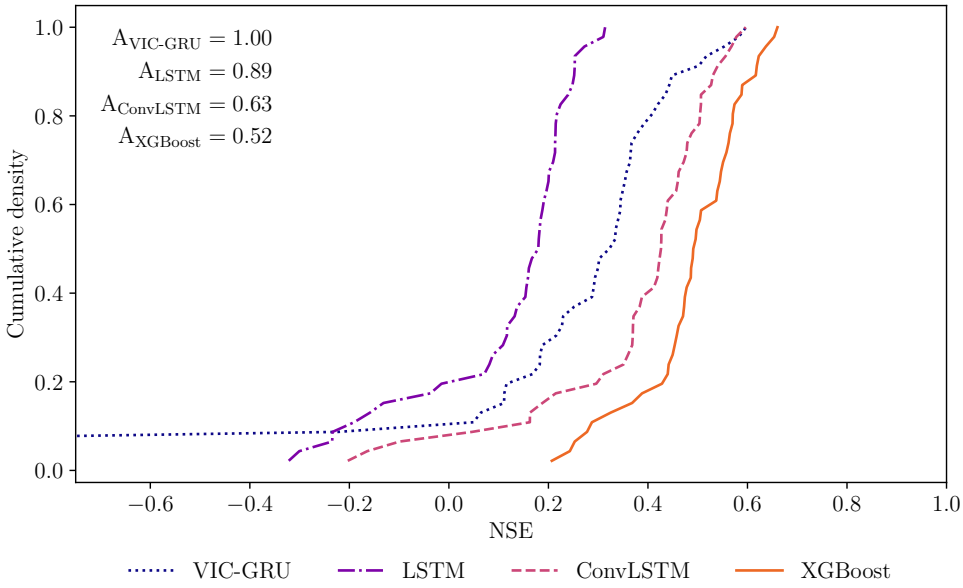


Figure 4.2: Cumulative NSE distributions for the physically-based VIC-GRU (dotted line), data-driven LSTM (dashed-dotted), ConvLSTM (dashed), and XGBoost (solid) models across the Lake Erie basins. The models are trained on all stations combined. A_* denotes each model's area under the distribution curve; lower values are better.

For basin 04207200 in Bedford, OH, USA (Figure 4.3, top), both VIC-GRU and XGBoost result in rather accurate predictions, with NSEs of 0.44 (VIC-GRU) and 0.57 (XGBoost). At basin 04177000 in Toledo, OH, USA (Figure 4.3, middle), VIC-GRU yields an NSE of 0.33 and outperforms XGBoost (0.13). The reason for this are a few days at the end of 2013 and in August 2014, where XGBoost overestimates the streamflow. The NSE metric is sensitive to such outliers, which is why the overall NSE is much worse than VIC-GRU's. Finally, at basin 04166500 in Detroit, MI, USA, VIC-GRU results in an NSE of -1.95, as it consistently overestimates flows, especially during the winter and spring months. XGBoost predicts these peaks more accurately and thus achieves a much higher NSE of 0.64. Notably, this basin is located in the metropolitan Detroit area and therefore represents a highly urbanized watershed with strong patterns of human water management.

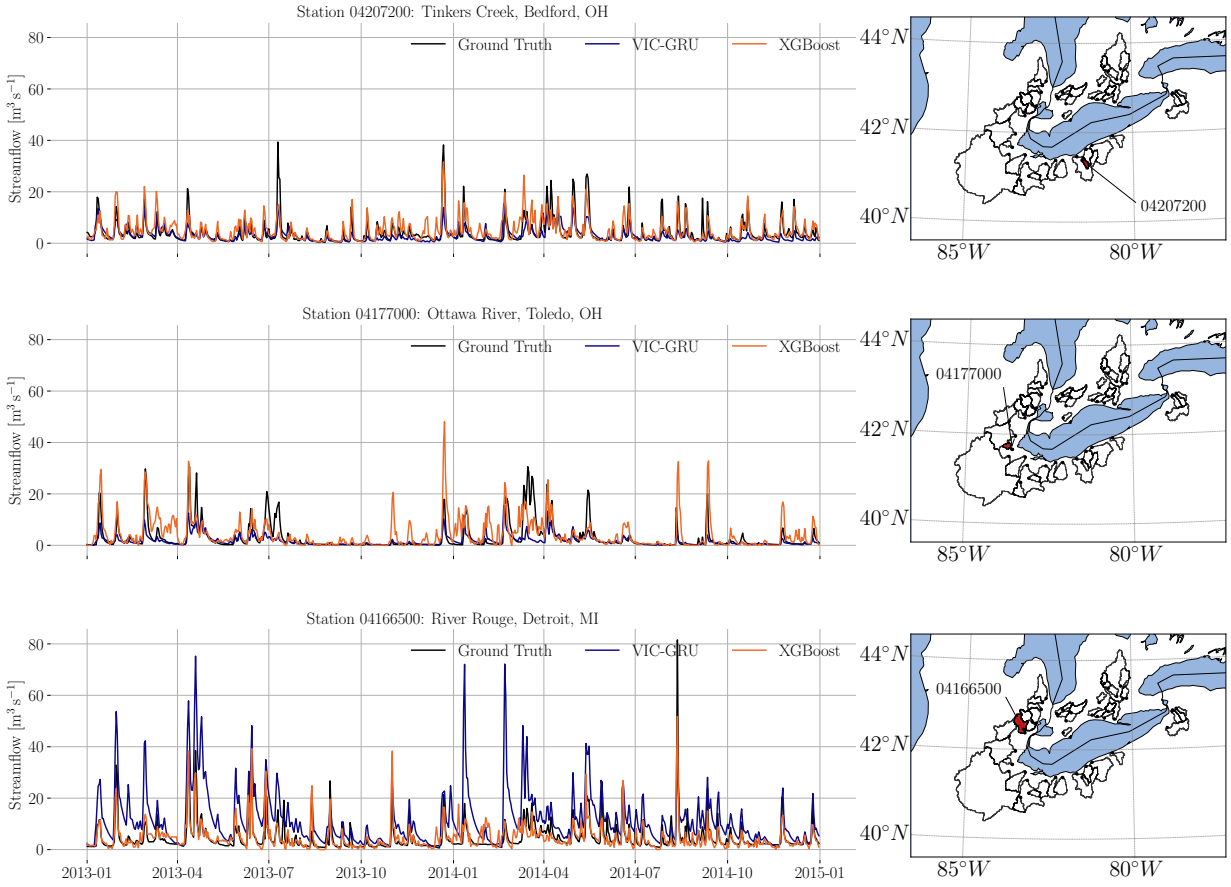


Figure 4.3: Time series of actual streamflow (black) and predictions for VIC-GRU (blue) and XGBoost (orange) at gauging stations 04207200, 04177000, and 04166500 during the test period 2013–2014.

4.4 Discussion

Although only a small case study, our results show that data-driven models can provide more accurate streamflow predictions than a physically-based model. Even to researchers who remain reluctant to adopt machine-learned models for streamflow prediction, our experiments at least reveal that there is more information in the input data than what the current physically-based models extract. A further advantage of data-driven models is their greater flexibility, which allows them to learn patterns of human water management. On the contrary, the assumptions that are inherent to physically-based models deprive them of such adaptiveness.

The LSTM's comparatively poor performance suggests that the Lake Erie dataset is too small to fully exploit the potential of data-driven models that operate directly on gridded data. This motivates our approach in the following case study, where we not only use a larger dataset, but we also aggregate the data for each basin to train lumped models. As such lumped models can take basin characteristics as input, we can still train a single model to predict arbitrary basins. In fact, it is architecturally easier to design lumped models that achieve such spatial generalization, since these models do not have to cope with differently-shaped gridded input fields that naturally stem from the basins' outlines.

Chapter 5

Hybrid Physically-Based and Data-Driven Streamflow Prediction

Even though our analyses in the previous chapter, together with other studies, provide ample evidence that data-driven models can outperform physically-based models, the hydrologic community continues to be hesitant about adopting such models, both in research and operationally [34]. To justify this reluctance, researchers and engineers often argue that machine-learned models lack interpretability and therefore trustworthiness, as they do not explicitly simulate hydrologic storages and states.

As a first step to overcome these issues, we propose a hybrid model that combines a physically-based model's interpretability with the predictive power of machine learning: We train a data-driven LSTM to predict the time series of differences between ground truth streamflow and the physically-based LBRM's predictions.

We evaluate our model as part of the second phase of the Great Lakes Intercomparison Project, which is still ongoing. Following up on lessons learned from the first project phase on Lake Erie, the second phase uses a larger forcing dataset and more carefully chosen basins that distinguish catchments of low human impact from most-downstream basins, which are the basins that correspond to the last gauging station before a stream flows into one of the lakes. Unlike low-impact basins, most-downstream basins can contain sub-basins in urban areas or show other types of human water management. Further, the second phase also performs spatial validation, where we test the models on previously unseen basins.

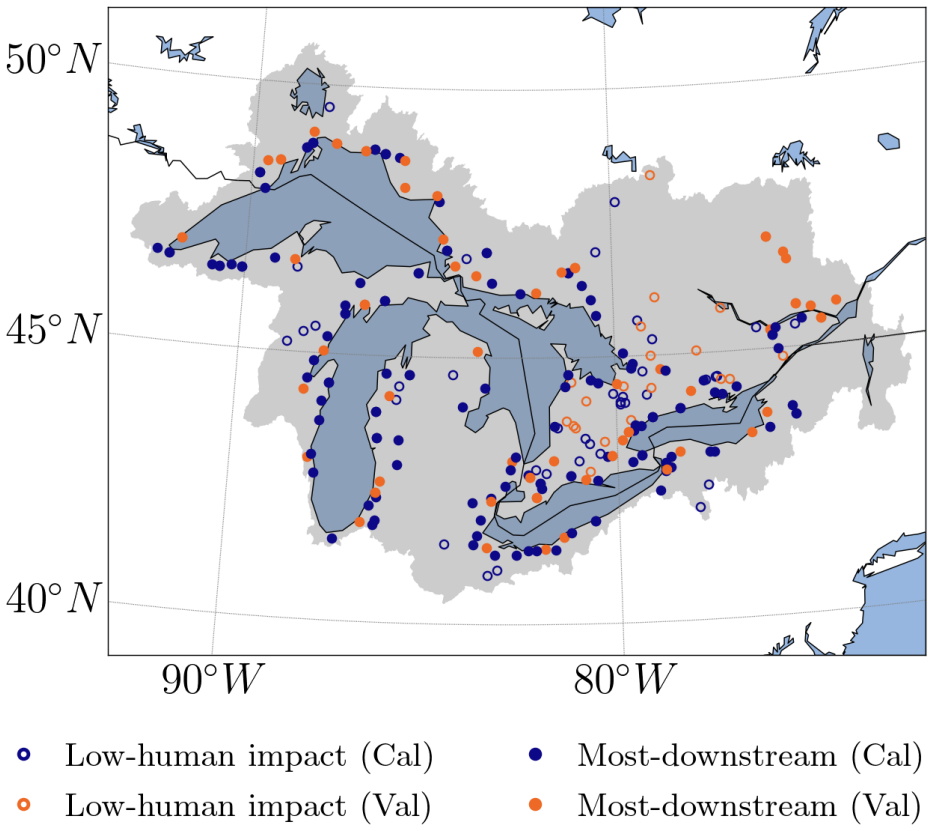


Figure 5.1: Map of the Great Lakes gauging stations in our analysis, divided into basins of low human impact (circles) and most-downstream basins (points) as well as calibration (blue) and validation (orange) basins.

5.1 Data

Figure 5.1 shows a map of the gauging stations we use in this case study. Improving on the previous study, we distinguish 99 basins of low human impact (i.e., not in highly urbanized areas and no human water management) and 156 basins that are the most downstream. In addition, we divide the basins into 141 calibration and 71 validation basins, and only use the calibration basins during training. The validation basins simulate predictions on ungauged basins, where we apply models to basins on which they have not been trained. Table 5.1 lists the resulting number of calibration and validation basins in the groups of low-impact and most-downstream basins.

Table 5.1: Number of calibration and validation basins, divided into low-impact and most-downstream basins. Since basins can be low-impact and most-downstream at the same time, the rows and columns do not add up to the total number of basins.

	Calibration	Validation	**Total**
Low-impact	66	33	**99**
Most-downstream	104	52	**156**
Total	**141**	**71**	**212**

Table 5.2: Meteorological forcing variables from the WFDEI-GEM-CaPA data product. Each variable covers the Great Lakes at a resolution of around 10 km for the years 2000 to 2016 in three-hourly time steps. The variables are available at the indicated vertical levels.

Variable	Explanation	Level	Unit
`PRECIP`	Quantity of precipitation	surface	mm
`TEMP_DAILY_AVE`	Average air temperature	40 m	°C
`TEMP_MIN`	Minimum air temperature	40 m	°C
`TEMP_MAX`	Maximum air temperature	40 m	°C

The meteorological forcing dataset we use in this study is a subset of the WFDEI-GEM-CaPA product [3], which contains precipitation and temperature information at different vertical measurement levels (see Table 5.2) for the Great Lakes region at a resolution of 10 km from 2000 to 2016. We use the first eleven years for training and evaluate on the remaining six years, both on the calibration basins we used during training (this is known as temporal validation in hydrology) and on the unseen validation basins (known as spatial validation in hydrology).

In addition, we provide the models with the static basin attributes described in Table 5.3. As we train all models once for all calibration basins combined, these basin characteristics allow the models to distinguish between different basins. Hence, we no longer feed the models one-hot-encoded basin identifiers.

As we will show in Chapter 6, a training set of eleven years and 141 basins is not much, even for lumped models; hence, we believe that the available data in this case study would not be sufficient to thoroughly train models directly on the higher-dimensional gridded data. Before training our models, we therefore aggregate the forcing data as well as static basin attributes for each basin into lumped time series.

Table 5.3: Static basin characteristics we use in the Great Lakes case study.

Variable	Explanation	Unit
`Area2`	Basin area	m^2
`RivSlope`	Mean river slope	%
`RivLen`	River length	km
`BasinSlope`	Mean basin slope	%
`BkfWidth`	Mean river bank width	m
`BkfDepth`	Mean river depth	m
`MeanElev`	Mean basin elevation	m
`FloodP_n`	Manning's river flood plain coefficient	$s/m^{1/3}$
`Q_mean`	Mean discharge	$m^3\,s^{-1}$
`CH_n`	Manning's river channel coefficient	$s/m^{1/3}$
`Perim_m`	Basin perimeter	m
`Regulation`	Indicator (0/1) of human regulation	–

5.2 Models

We compare LBRM, an existing lumped physically-based model, with XGBoost, a standard LSTM architecture, and a combined model that uses an LSTM to predict the error of the physically-based LBRM model.

LBRM. The Large Basin Runoff Model (LBRM) as introduced in Section 2.3 is a lumped physically-based model with six states [9]. This model takes precipitation and minimum and maximum temperature as input. Unlike the other models in this case study, LBRM is trained on each basin individually. As of yet, we only have LBRM predictions for most calibration stations, but not for validation stations; hence, we only perform temporal evaluation for the LBRM model.

XGBoost. We feed the XGBoost model a fixed-history window of the previous 30 days' precipitation, average, minimum, and maximum temperature, concatenated with the static basin attributes into a flattened vector. Further, we normalize the input variables and the ground truth to a mean of zero and standard deviation of one. As loss function, we use NSE' (cf. Section 2.2.1), and we perform a cross-validated random search to find suitable model hyperparameters. To reduce errors due to random initialization, we train each architecture with eight different seeds and evaluate the

ensemble of their averaged predictions. Appendix A provides more details on the parameter tuning and training procedures.

LSTM. The LSTM model consists of one 256-neuron hidden layer with a dropout rate of 0.4. We train the model on the last 270 days' forcings, however, we concatenate the static basin attributes to each forcing time step, as LSTMs support time-series input. The reason why we choose a history of 270 days for the LSTM and 30 days for XGBoost will become clear in Chapter 6, where we examine the optimal input sequence lengths for both models. Again, we normalize the input and output variables to a mean of zero and a standard deviation of one, and the loss function is NSE′. As for XGBoost, we average the predictions of eight models with different seeds. Appendix A provides further details on our model setup and training procedures.

LBRM+LSTM. Our hybrid model consists of a physically-based LBRM and a data-driven LSTM component. The LBRM component takes as input the meteorological forcings and generates a time series of predicted states for each time step (snow pack, upper and lower soil moisture, groundwater, surface storage, and streamflow). Subsequently, the LSTM component ingests the meteorological forcings and LBRM's predicted states at each time step and predicts the difference between ground truth and LBRM's streamflow prediction. To obtain the final predictions, we add the LSTM's error predictions to the LBRM predictions. The architecture and training procedure of the LSTM are identical to those of the pure LSTM model. As we are still awaiting LBRM predictions for the basins in the spatial validation set, we only perform temporal evaluation for this model.

We choose to evaluate a combination of LBRM and LSTM rather than, for instance, a combination of LBRM and XGBoost, since we believe the former to be more promising. The reasons for this are two-fold: First, we will show that the pure LSTM already results in better predictions than XGBoost due to the larger training set as compared to our study on Lake Erie. Therefore, it might more successfully extract additional information from LBRM's states. Second, since XGBoost does not directly ingest time series, we would need to flatten the input sequence of LBRM states together with forcings and static basin attributes into a single vector. With six LBRM states and sequences of length 30, this would add $6 \times 30 = 180$ dimensions to the input space. We consider it likely that the available dataset is insufficient to adequately train a model on this space.

Table 5.4: Minimum, median, and maximum of the NSE distributions for the physically-based model LBRM, the two data-driven models XGBoost and LSTM, and the combined model LBRM+LSTM on the Great Lakes basins. Section A shows the results for temporal validation, while section B refers to spatial validation on low-impact and most-downstream basins (at this point, we only have spatial validation results for XGBoost and LSTM). Best values are highlighted in bold.

			Statistic	LBRM	XGBoost	LSTM	LBRM+LSTM
(A)	Temporal validation		Min	−24.16	−0.27	0.08	**0.20**
			Median	0.41	0.37	0.60	**0.66**
			Max	0.82	0.65	0.82	**0.88**
(B)	Spatial validation	low-impact	Min	–	**−0.41**	−1.95	–
			Median	–	0.37	**0.51**	–
			Max	–	0.57	**0.78**	–
		most-downstream	Min	–	−8.03	**−1.99**	–
			Median	–	0.27	**0.38**	–
			Max	–	0.52	**0.70**	–

5.3 Results

Table 5.4A lists each model's minimum, median, and maximum NSE in temporal validation (i.e., predictions on the training basins for the test period). Table 5.4B shows the spatial validation results (i.e., predictions on previously unseen basins) for the purely data-driven models. Naturally, the accuracy in spatial validation is lower than in temporal validation, as the models have not seen these basins during training. Among the purely data-driven models, the LSTM mostly outperforms XGBoost, as it shows higher minimum, median, and maximum NSE values in all cases except spatial validation on low-impact basins, where XGBoost's minimum NSE is better. This is in contrast to our results on Lake Erie, likely due to the larger training set and the lower-dimensional lumped setup. The physically-based LBRM model has lower minimum, median, and minimum NSE values than the LSTM. Compared to XGBoost, LBRM has higher median and maximum accuracy but a far worse minimum NSE value. The combined LBRM+LSTM model yields the best minimum, median, and maximum NSE values of all models in temporal validation.

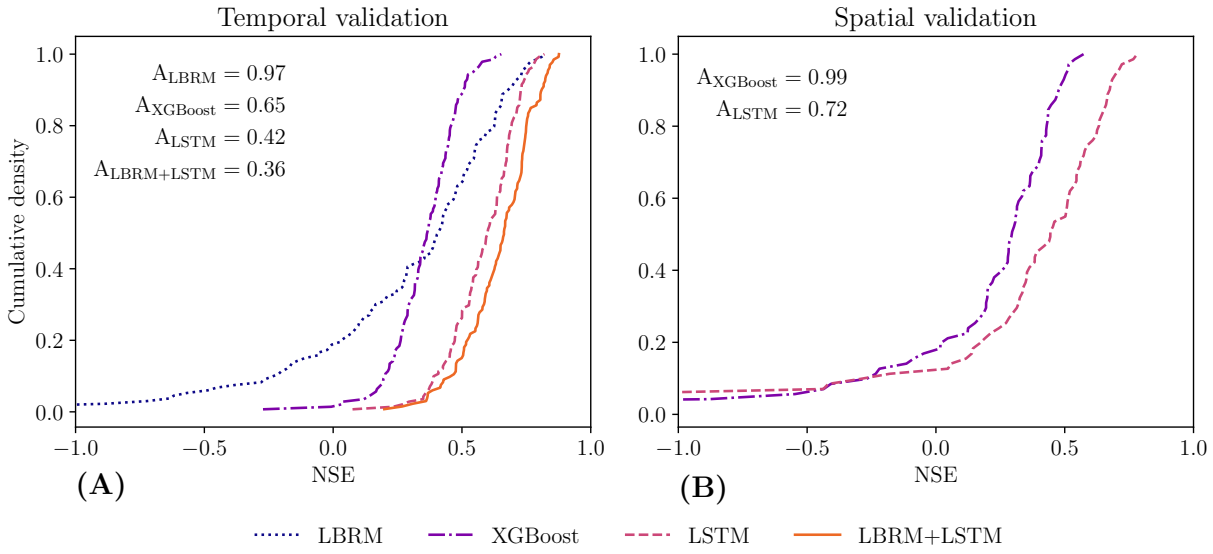


Figure 5.2: Cumulative NSE distributions for the physically-based LBRM (dotted line), data-driven XGBoost (dashed-dotted), LSTM (dashed), and combined LBRM+LSTM (solid) models across the basins in temporal validation (panel A) and the basins in spatial validation (panel B). A_* denotes each model's area under the distribution curve; lower values are better.

Figure 5.2A provides a more detailed view on these results, as it shows the models' cumulative NSE distributions across the training basins and denotes the corresponding area A under the distribution curve. Visually, worse models exhibit curves that are shifted leftwards; quantitatively, this expresses in larger areas A under the distribution curve. We see that the LBRM distribution has a much heavier left tail of poorly predicted basins than all other models, even though it outperforms XGBoost on the basins it predicts well. The LBRM+LSTM model provides the best NSE values, resulting in the lowest area A of 0.36—in other words, the combination of LBRM and LSTM is more accurate than either model individually on most basins.

Figure 5.2B shows the cumulative distributions for XGBoost and LSTM across all spatial validation basins (both low-impact and most-downstream). Consistent with Table 5.4, the models' results are worse than in temporal validation: For XGBoost, the area under the curve increases from $A_{\text{XGBoost}} = 0.65$ to 0.99, and for the LSTM from $A_{\text{LSTM}} = 0.42$ to 0.72. Still, the LSTM performs better than XGBoost on most basins.

Figure 5.3 shows the hydrographs of LBRM and LBRM+LSTM as well as the LSTM component's predicted corrections at three exemplary basins. Gauging station 04085427 near Manitowoc, WI, USA (Figure 5.3, top), is one of the few examples where the hybrid model slightly deteriorates the prediction quality compared to LBRM, from an NSE of 0.71 to 0.67. This represents the largest reduction in NSE among all basins in our study. It appears that in this case, the LSTM component lessens LBRM's relatively accurate peak predictions during the snow melt seasons. In contrast, station 04126970 near Mayfield, MI, USA (Figure 5.3, middle), experiences the largest NSE improvement in our study: The pure LBRM is completely off throughout the entire test period, as it overestimates modest flows by about $6\,\mathrm{m}^3\,\mathrm{s}^{-1}$ and does not properly anticipate peak flows (NSE -24.16). We speculate that this might be due to an error in calibration. The added LSTM component corrects the continual bias and better predicts high flows, which improves the NSE to 0.73. Lastly, station 04119000 near Grand Rapids, MI, USA (Figure 5.3, bottom), is an example where the LSTM component improves already good LBRM predictions. LBRM achieves an NSE of 0.75, which the added LSTM raises to 0.88—the best NSE in our experiments. Based on visual inspection, this improvement stems from better adaption to peak flows—especially during an extreme event in 2013—and from more precise modeling of the decay after such peak flows.

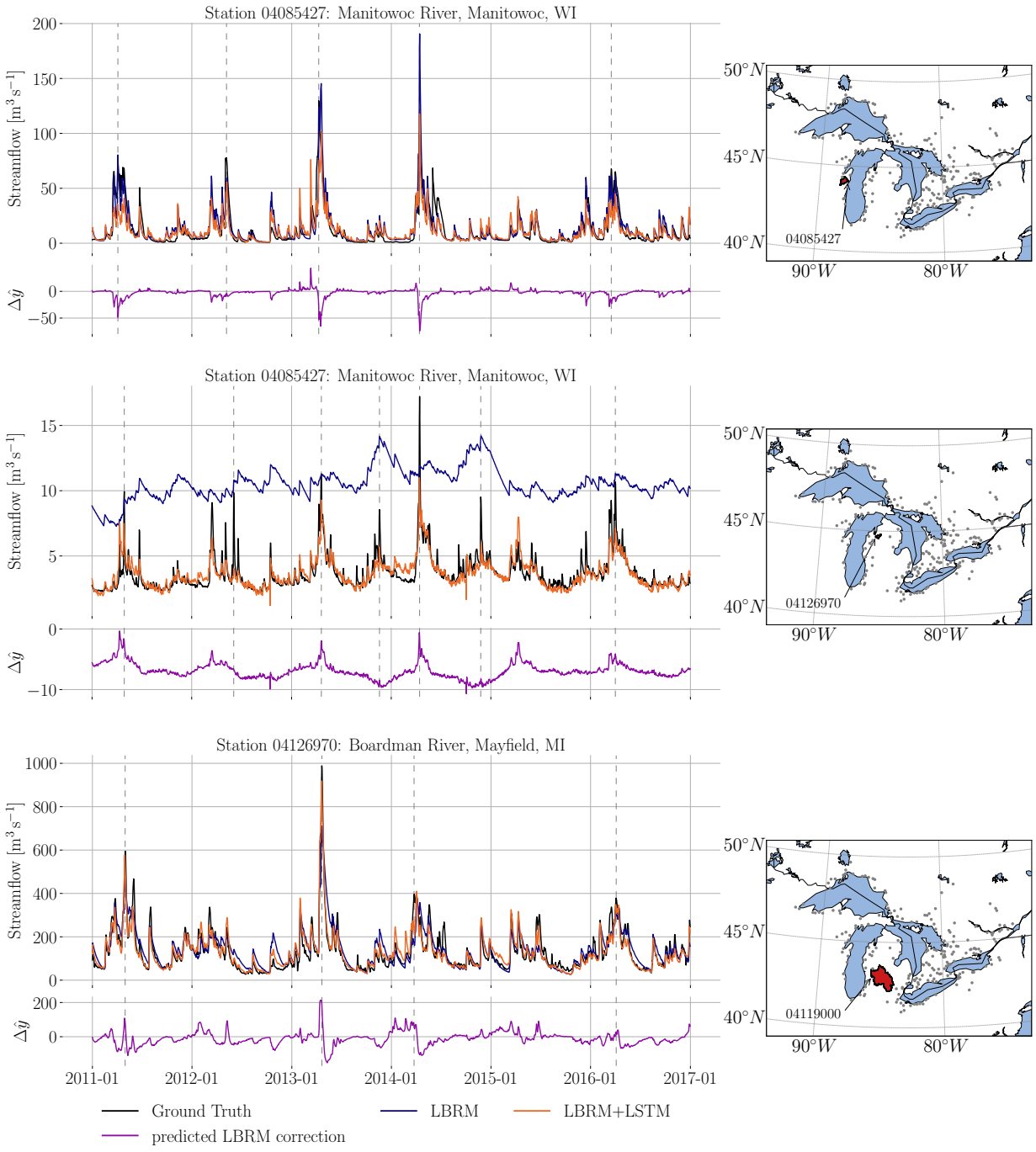


Figure 5.3: Time series of actual streamflow (black) and predictions for LBRM (blue) and LBRM+LSTM (orange), and predicted corrections (purple) at gauging stations 04085427, 04126970, and 04119000 in the test period 2011–2016. Dashed lines highlight peak flows.

5.4 Discussion

Consistent with the results of our case study on Lake Erie, data-driven models outperform the physically-based LBRM model. Although LBRM has higher NSE values than XGBoost on some basins, the data-driven models' prediction quality is more consistent, as the shorter left distribution tails and smaller areas in Figure 5.2 show. As expected, the LSTM benefits from the additional training data and the lumped setup. Consequently, it provides predictions that are not only more accurate than LBRM and XGBoost, but also more accurate than the LSTM in the first case study. The fact that the LSTM's accuracy on low-impact *test* basins is higher than LBRM's accuracy on the *training* basins indicates that the LSTM would outperform LBRM in spatial validation, too.

We believe that the outstanding results for the combined LBRM+LSTM model are promising precursors of joint data-driven and hydrologic modeling efforts. The hybrid model predicts peak and base flows with high accuracy and improves the predictions of decay after high-flow events. While one might argue that the LSTM's substantial changes to the LBRM output render any interpretation of LBRM states futile, we argue that the opposite is true: The meaningfulness of purely physically-based models' interpretations is doubtful in situations where the models' predictions are wrong. To this end, the combination with data-driven models yields a notion of the interpretations' trustworthiness—the states can only be reliably interpreted if the error correction is small. If, on the other hand, the data-driven error prediction is large, this means that the physically-based states are likely incorrect, which in itself can be a helpful interpretation that hints towards ways of improving the model.

To take this approach even further, we can imagine future work that uses data-driven models such as combinations of Gaussian processes and neural networks to explicitly predict both the physically-based error and confidence intervals [11].

Chapter 6

Effects of Limited Training Data

As we demonstrated in the previous chapters' case studies, data-driven models can outperform physically-based models. We also saw, however, that the quality of data-driven predictions seemed limited by the available training data. While the LSTM performed poorly in the Lake Erie case study, it outperformed XGBoost and LBRM on the larger Great Lakes dataset, and it might become even more accurate given more training data. In this chapter, we clarify the models' qualities on a large and diverse dataset of basins with known streamflow measurements. We use 19 years of historical records from over 500 basins across the continental United States to quantify the effects of limited training data in terms of time and space on tree- and LSTM-based streamflow prediction models.

Our analyses answer the following two main research questions:

- When we train a model on a given number of training years, we can still choose how many previous days of forcings we feed the model to generate a single prediction—the input sequence length. Which sequence length yields the best predictions for each model, and does this depend on the training set size?
- How do training period length and the number of basins in the training dataset affect the prediction quality of tree- and LSTM-based models?

6.1 Data and Method

6.1.1 Data

For our experiments, we use the *Catchment Attributes and Meteorology for Large-sample Studies* (CAMELS) dataset, provided by the U.S. National Center for Atmospheric Research [36]. This dataset contains static attributes, streamflow measurements, and three different lumped meteorological time-series products for 671 basins across the continental United States. A number of studies use it as a sufficiently large and diverse dataset; hence, we believe that our results generalize to other regions and datasets [1, 25, 33, 35]. Following several of these studies, we use the Maurer forcing product (which is one of the three products in the CAMELS dataset) [31]. Table 6.1 lists the meteorological attributes from this dataset that we use in our study. Appendix B provides a list of the static basin features contained in the CAMELS dataset. Following previous studies on the same dataset, we discard basins with high discrepancy between their area as calculated using different methods and train and test our models on the remaining 531 basins [25, 35].

Table 6.1: Maurer meteorological variables in the CAMELS dataset used in this study.

Variable	Explanation	Unit
`prcp`	Daily cumulative precipitation	mm
`tmin`	Daily minimum air temperature	°C
`tmax`	Daily maximum air temperature	°C
`srad`	Average short-wave radiation	$\mathrm{W\,m^{-2}}$
`vp`	Vapor pressure	Pa

6.1.2 Training and Evaluation Procedures

As data-driven models, we use XGBoost and the Entity-Aware LSTM (EA-LSTM) model proposed by Kratzert et al. [25] as introduced in Section 2.4. To reduce errors due to random initialization, we train each architecture with eight different seeds and evaluate the ensemble of their averaged predictions. We use NSE' (cf. Section 2.2.1) as the loss function for all models. Appendix A provides further details on our training procedures for the two types of models.

To determine the effect of limited training data on the models' prediction quality, we vary the training set size with respect to the following three dimensions:

Training period length. The amount of historical data for a given basin. This is perhaps the most obvious and most commonly exploited dimension in streamflow prediction.

Hydro-geo-climatic diversity. The number of basins that we use to train a single model. While the advantage of greater geographic, hydrologic, and geophysical diversity may at first seem counter-intuitive, we hope that models are able to generalize knowledge across basins. For instance, a model may have never seen a period of drought in a certain basin during training, but it may have seen a drought in a basin with similar characteristics. In such a case, an ideal model would transfer the knowledge from the second basin to the first and yield accurate predictions, even though it was never trained to predict droughts on the first basin.

Input sequence length. Training period length and hydro-geo-climatic diversity determine the overall number N of observations that we show the models during training. Additionally, we can vary the amount of data we feed a model to predict an individual sample: when predicting streamflow y_t for day $t \in [1, \dots, T]$, we feed the model the previous k days' forcings $x_{t-k+1}, \dots, x_t$. This parameter k is called the input sequence length; it controls the amount of data we feed the models to fit an individual streamflow observation y_t. Note that, unlike the other two dimensions, the sequence length is independent of the overall dataset size (apart from the training period length being an upper bound). Rather, it is a tunable parameter of model training; for instance, too short sequences (small k) will not contain sufficient information for accurate predictions, and too long sequences (large k) make it challenging to identify the most important patterns.

We use input sequence lengths k of 10, 30, 100, 270, and 365 previous days' forcings for EA-LSTMs. For XGBoost, we feed the sequences of length k as flattened vectors of $5k$ variables (precipitation, minimum and maximum temperature, radiation, and pressure), concatenated with the 27 basin attributes (see Appendix B) to a vector of length $5k + 27$. We do not use sequences longer than 100 days for XGBoost, as we find that the experiments with $k = 100$ already result in worse predictions than lengths of 30. Likely, this is because the sequence length affects the input dimension by a multiplicative factor, as we use a flattened vector as input. A history of 100 days already leads to a $100 \times 5 + 27 = 527$-dimensional input space, where training is challenging. Further, longer sequences drastically increase the runtime of the already computation-intensive hyperparameter search.

Besides the sequence length, we evaluate the quality of each model by varying the amount of training data in terms of the number of training years and the number of basins. All training periods start from October 1999 and last three, six, or nine years.

The basins are random subsets of 13 (2.5 %), 26 (5 %), 53 (10 %), 265 (50 %), and all 531 (100 %) basins. For the conditions that do not include all basins, we evaluate five different random basin selections each. Following the same setup as Kratzert et al., we test all models on the same test period from October 1989 until September 1999, and we evaluate them on the set of basins that they were trained on.

For each of the 15 combinations of three training period lengths and five basin set sizes, we select the EA-LSTM and XGBoost model trained with the input sequence length k that results in the best median NSE across all basins and across the five random basin sets.

As a result, we obtain two distributions $F_{\text{EA-LSTM}}$ and F_{XGBoost} of NSE values for the models. Each distribution has a sample size of five times the number of basins (except for the case of all 531 basins, where we do not have five random basin sets). We use a Kolmogorov–Smirnov significance test to assess the null hypothesis that the distributions of NSE values are identical. For this, we use the scipy function `scipy.stats.ks_2samp`. This test is based on the maximum absolute difference between $F_{\text{EA-LSTM}}$ and F_{XGBoost}. The significance test's p-value denotes the probability that the NSE values are at least as different as in our experiment even though they come from the same distribution. To account for the large number of significance tests (15), we apply Bonferroni correction [32] and test our hypotheses at $\alpha = 0.01/15$. Hence, we reject the null hypothesis of identical distributions for p-values above $0.01/15$ and accept it otherwise. Further, we estimate the corresponding effect size as Cohen's d [8]. The effect size is a metric for the difference of distributions; it measures the difference between the means of the two NSE distributions, normalized by their combined standard deviation ($d = (\mu_{\text{EA-LSTM}} - \mu_{\text{XGBoost}})/\sigma_{(\text{EA-LSTM,XGBoost})}$). The larger d, the further apart are the distribution means.

While the effect size d measures the difference between the two models' NSE distributions, we quantify the quality of each model's predictions as the area A under the curve of its cumulative NSE distribution. Ideally, this value is zero, with an NSE of one on all basins. As models yield smaller NSEs, the distribution shifts leftward and A increases. Larger values of A therefore correspond to worse overall performance.

6.2 Results

6.2.1 Input Sequence Length

Table 6.2A shows the best sequence lengths for XGBoost and EA-LSTM. As expected, the best sequence length generally increases with dataset size. For XGBoost, input sequence lengths beyond 30 perform worse even on larger training sets, likely due to the

Table 6.2: Input sequence lengths k for XGBoost and EA-LSTM that yield the best median NSE for each number of basins (rows) and training years (columns) (section A). Section B shows the difference in median NSE between the best and next-smaller sequence length. The cell entries are color-coded; lighter colors correspond to larger values.

				Training years					
				XGBoost			EA-LSTM		
				3	6	9	3	6	9
(A)	Number (percentage) of basins	13	(2.5 %)	10	10	10	10	10	30
		26	(5 %)	10	30	30	10	30	30
		53	(10 %)	10	30	30	30	30	100
		265	(50 %)	30	30	30	100	270	365
		531	(100 %)	30	30	30	270	365	365
(B)	Number (percentage) of basins	13	(2.5 %)	–	–	–	–	–	0.046
		26	(5 %)	–	0.022	0.044	–	0.048	0.058
		53	(10 %)	–	0.033	0.045	0.044	0.063	0.014
		265	(50 %)	0.053	0.062	0.070	0.017	0.005	0.002
		531	(100 %)	0.061	0.072	0.078	0.007	0.005	0.001

aforementioned multiplicative increase of training space dimensionality. For EA-LSTMs, the optimal sequence length continues to grow with larger datasets, reaching the longest evaluated length of 365 on the three largest training sets.

Table 6.2B shows the difference in median NSE between the best and next-smaller input sequence length for each training set size. The difference increases for XGBoost (likely because of its constant sequence length of 30 on larger training sets) but decreases for the EA-LSTM models, where it almost reaches zero on the largest configuration.

6.2.2 Training Period Length and Number of Basins

Table 6.3 provides a tabular comparison of the models' minimum, median, and maximum prediction quality on the different training set sizes, as well as the average percentage of basins with NSEs below zero. The given results correspond to the models that use their respectively optimal input sequence length k (see Table 6.2A). Figure 6.1 visualizes

Table 6.3: Minimum, median, and maximum NSE scores and average percentage of failed basins (p_{failed}[%], NSE ≤ 0) on the test period (Oct. 1989 to Sep. 1999) for XGBoost and EA-LSTM models, trained with different amounts of training years (N_{years}) and basins (N_{basins}). The values are calculated across five different random basin selections (we aggregate minimum as minimum, median as median, maximum as maximum, and p_{failed} as the average percentage of failed basins in each random basin set). In each row, the best values for each metric are highlighted in bold.

N_{years}	N_{basins}	XGBoost				EA-LSTM			
		Min	Median	Max	p_{failed}[%]	Min	Median	Max	p_{failed}[%]
3	13	−0.64	0.41	0.69	4.62	**−0.31**	**0.43**	**0.70**	**1.54**
	26	−4.71	0.43	**0.81**	4.62	**−1.58**	**0.48**	0.80	**3.08**
	53	−1.98	0.48	0.85	2.26	**−1.36**	**0.56**	**0.88**	**0.38**
	265	−1.92	0.55	0.84	1.81	**−0.25**	**0.65**	**0.91**	**0.30**
	531	−1.34	0.57	0.87	1.51	**0.03**	**0.68**	**0.93**	**0.00**
6	13	**−1.06**	0.49	0.72	4.62	−1.65	**0.57**	**0.77**	**1.54**
	26	−4.14	0.53	0.78	3.08	**0.01**	**0.61**	**0.84**	**0.00**
	53	−1.15	0.58	0.86	0.75	**−0.26**	**0.64**	**0.91**	**0.38**
	265	−3.08	0.63	0.90	1.36	**−0.20**	**0.71**	**0.94**	**0.45**
	531	−1.75	0.64	0.91	0.94	**−0.90**	**0.72**	**0.95**	**0.56**
9	13	−1.39	0.54	0.75	3.08	**−0.16**	**0.63**	**0.81**	**1.54**
	26	−2.41	0.57	0.81	2.31	**−0.01**	**0.64**	**0.87**	**0.77**
	53	−1.22	0.61	0.89	1.13	**−0.02**	**0.68**	**0.93**	**0.38**
	265	−4.91	0.65	0.91	1.43	**−0.96**	**0.73**	**0.95**	**0.23**
	531	**−1.21**	0.66	0.91	1.13	−1.38	**0.74**	**0.96**	**0.19**

these results in cumulative distribution plots. Each plot in a particular row and column in Figure 6.1 corresponds to one of the 15 combinations of basin set size and training period length, and it shows the two models' empirical cumulative NSE distributions when we use the respectively best input sequence length (cf. Table 6.2A). An ideal model in this figure would exhibit "⌟"-shaped curves, with an NSE of one on all basins. Quantitatively, this would correspond to an area under the distribution curve of $A = 0$. Further, we highlight configurations where the Kolmogorov–Smirnov test reports that the distributions of XGBoost and EA-LSTM differ significantly with an effect size $d > 0.35$ (this threshold is halfway between Sawilowsky's suggestion for "small" and "medium" effect sizes [40]).

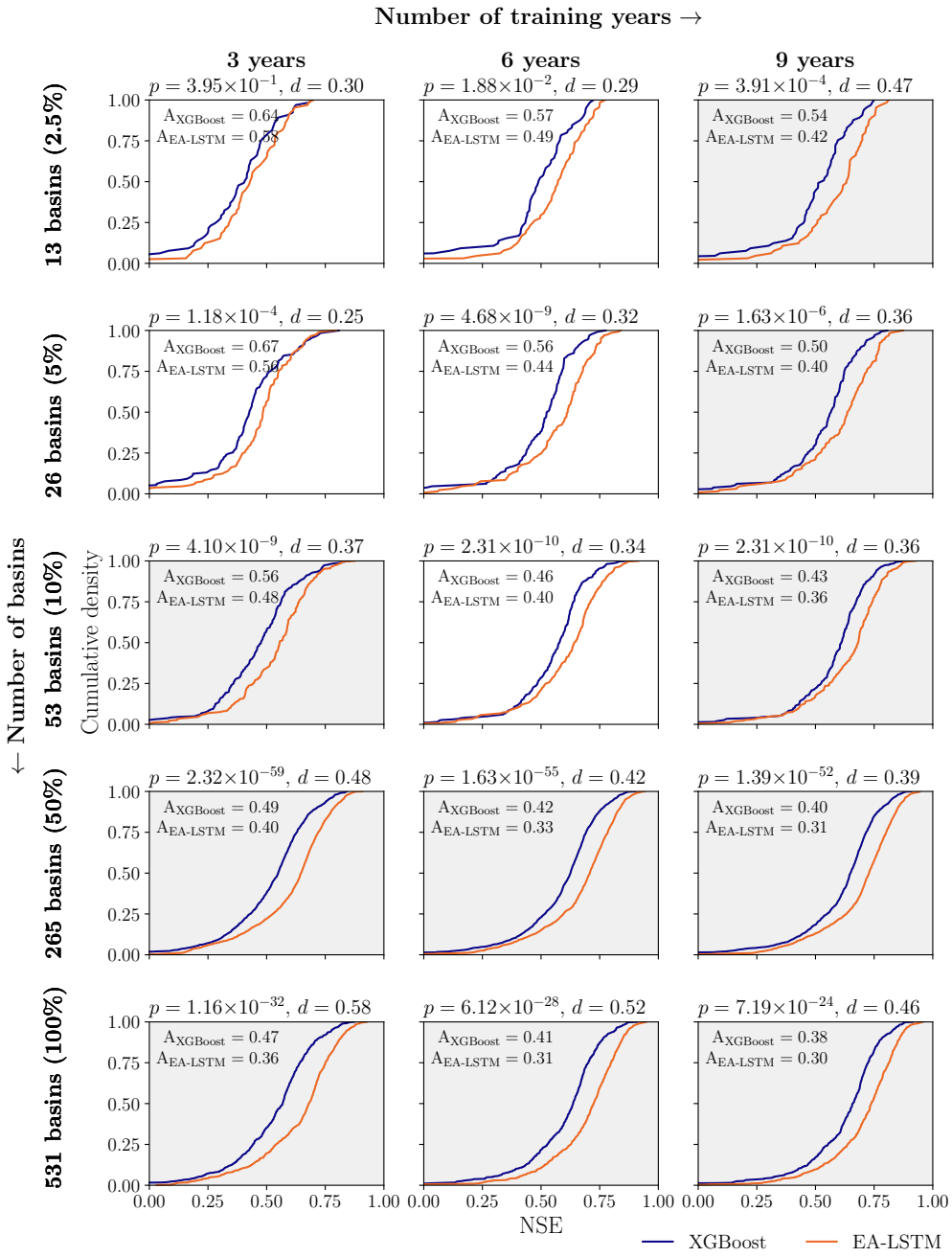


Figure 6.1: Cumulative NSE distributions for XGBoost (blue) and EA-LSTM (orange) at varying training set sizes in terms of training period (columns) and number of basins (rows) using the optimal input sequence length k (cf. Table 6.2A). Each plot shows the p-value of a Kolmogorov–Smirnov significance test and the effect size as Cohen's d. Plots with gray backgrounds correspond to combinations with $p < 0.01/15$ and $d > 0.35$, which indicates distribution pairs with at least small to medium differences. A_{XGBoost} and $A_{\text{EA-LSTM}}$ indicate the area under the distribution curves; lower values are better.

The plots show that, on very limited data (three to six years and 13 basins), the distributions for the two models do not differ significantly, although the EA-LSTM's median NSE is slightly higher. As expected, more training data improves the predictions. Visually, this reflects in the rightward shift of the distributions in Figure 6.1. Quantitatively, the area under the curves decreases from $A_{\text{XGBoost}} = 0.64$, $A_{\text{EA-LSTM}} = 0.58$ on the smallest to $A_{\text{XGBoost}} = 0.38$ and $A_{\text{EA-LSTM}} = 0.30$ on the largest configuration. At the configurations with three years and 53 basins, at least 265 basins, or at least nine years, the EA-LSTM outperforms XGBoost with an effect size above 0.35. The effect size increases with the number of basins but does not show a clear correlation to the number of training years.

Figure 6.2A visualizes the relation between training set size and median NSE for XGBoost (circles) and EA-LSTM (squares) in a scatter plot. Lighter colors denote larger numbers of basins, whereas larger markers correspond to more training years. For all but the 531-basin configurations, the plot shows five points, which represent the five random basin subsets. As training set size increases, we see squares more consistently above circles of same color and size, which means that the EA-LSTM outperforms XGBoost with increasing consistency. Further, both models not only benefit from additional training years, but also from additional basins: When we keep the training years constant (markers of same size and shape) but increase the basin subset size (lighter colors), the predictions improve. Additional training years, however, seem to improve predictions more than additional basins (darker, larger markers are above lighter, smaller ones).

From a more high-level perspective, Figure 6.2B groups the median NSE values by the orders of training set size magnitude denoted by gray lines in Figure 6.2A. Consistent with the previous analyses, the models show similar prediction accuracy on the small training sets, but EA-LSTMs outperform XGBoost on larger training sets. Up to training set sizes of around $2^4 \times 10^4$ samples, we see strongly improving NSE scores. On even larger training sets, the NSEs continue to improve, but at a much slower pace.

Finally, Figure 6.3 visualizes the NSE distributions' spatial patterns in two heatmaps. Panel A shows the models' NSE values on the smallest training set, while panel B shows the results on the largest training set. XGBoost's predictions are generally slightly worse than the EA-LSTM's, but the accuracy of both models is strongly correlated: On the small training set (Figure 6.3A), we observe a Pearson correlation coefficient of 0.84; on the large dataset (Figure 6.3B), a correlation coefficient of 0.78. The models appear to make poor predictions on the same basins, with only two visually striking exceptions in Figure 6.3B: one where XGBoost results in much worse predictions than the EA-LSTM (NSE 0.29 vs. 0.81), and one vice versa (NSE 0.73 vs. 0.02). Manual examination reveals that these cases result from few days where one model largely over-predicts streamflow. The NSE metric's sensitivity to such outliers consequently deteriorates the score on the affected basins.

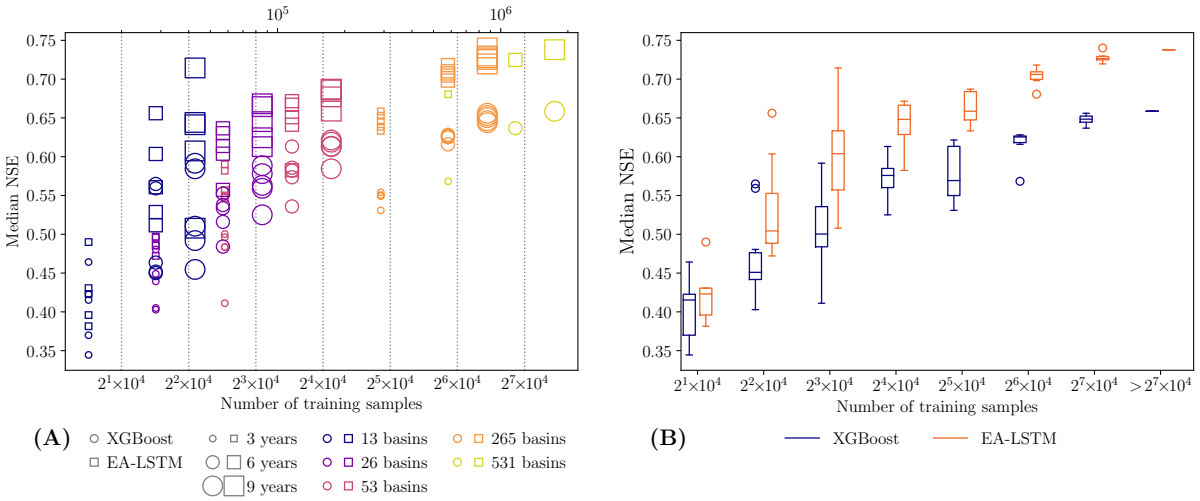


Figure 6.2: Relation between number of training samples ($N_{\text{basins}} \times (N_{\text{years}} \times 365)$) and median NSE across all basins. Markers in panel A represent the median NSE for XGBoost (circles) and EA-LSTM (squares) on a certain training period length (three to nine years, marker size) and number of basins (13–531, marker color). For the combinations with less than 531 basins, we report one median NSE score for each of the random basin subsets. Panel B aggregates these median NSEs for XGBoost (blue) and EA-LSTM (orange) into one boxplot per order of training set size magnitude as indicated by the gray lines in panel A. The boxes extend from lower to upper quartile and have a line at the median. The whiskers reach to the last data point that is up to $1.5 \times (p_{75} - p_{50})$ beyond the ends of the boxes; circles indicate outliers beyond those points. Note the logarithmic x-axis scales.

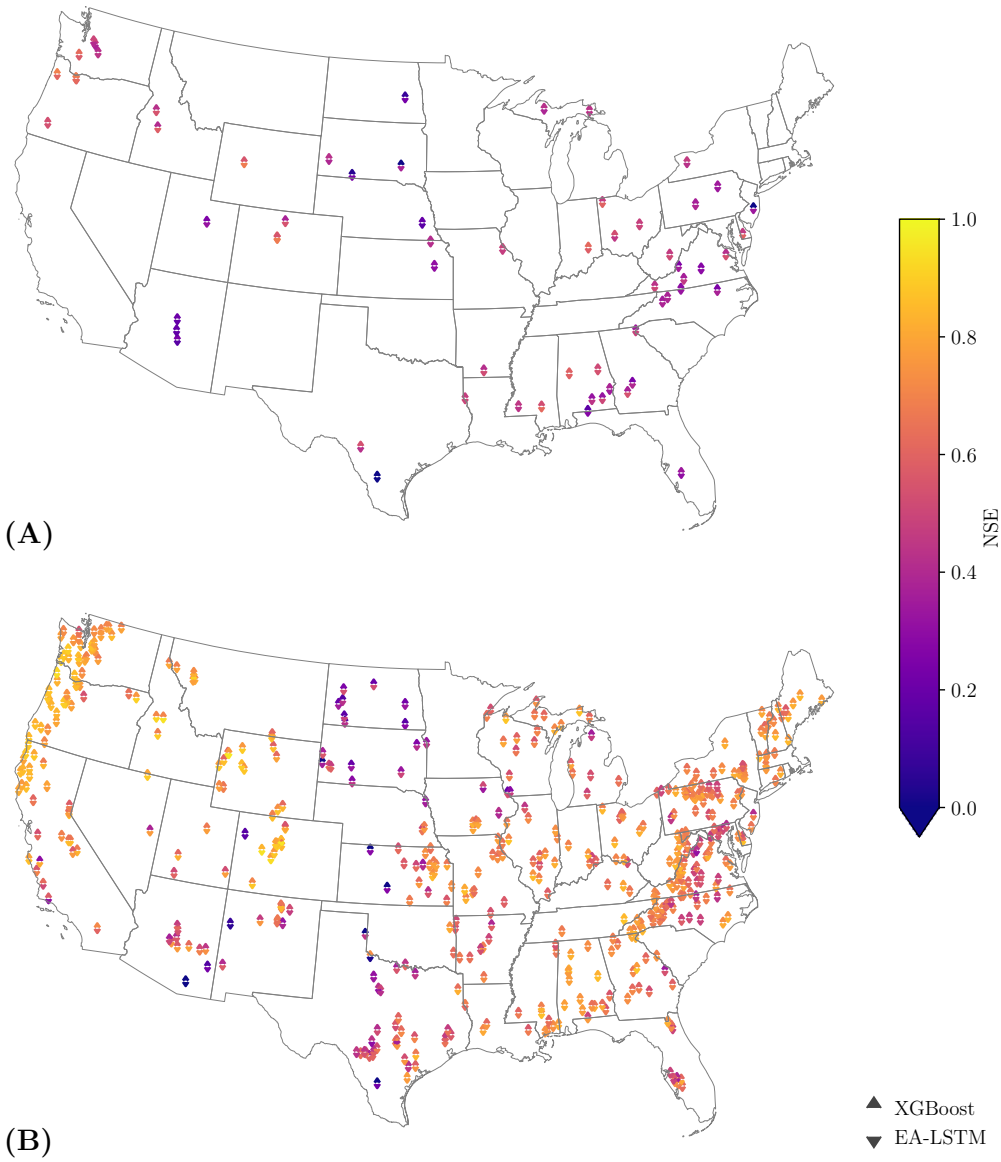


Figure 6.3: Heatmap visualization of the XGBoost and EA-LSTM models' NSE values for each basin. Panel A shows the NSE values on the smallest configuration of three years and 13 basins. It displays basins that are part of at least one random 13-basin subset. Where a basin is part of multiple subsets, we show the average NSE. Panel B shows the NSE scores on the largest configuration of nine years and all 531 basins.

6.3 Discussion

6.3.1 Input Sequence Length

Our findings reveal connections between the different dimensions of training set size: longer training periods and more basins call for longer input sequences. This becomes especially clear when we compare our results with those from a preceding study, where we trained models on fixed-length input sequences [12]. There, EA-LSTMs performed worse than XGBoost on small datasets, whereas the adaptive sequence lengths we present above improve the EA-LSTM's predictions to a similar accuracy as XGBoost's on small training sets.

The small differences in NSE between the sequence lengths of 270 and 365 for EA-LSTM on the largest training configurations (cf. Table 6.2B) indicate that even longer sequences would not further improve predictions. Also, this observation makes it seem unlikely that the optimal sequence length increases beyond 365 for even larger training sets. For XGBoost, the stable sequence length of 30 indicates that training sets would need to be substantially larger to successfully exploit longer sequences.

6.3.2 Training Period Length and Number of Basins

While it is not surprising that additional training years improve predictions, it was not clear from the outset that more basins yield this effect, too. Intuitively, one might think that a model that needs to predict fewer basins could better adapt to these basins' streamflow patterns—as we do not test the model on unseen basins, the model could overfit on the training basins. Yet, it turns out that this is not the case, as the models benefit from additional training years *and* additional basins. The growing effect size d and the decreasing area A under the cumulative distribution curve suggest that EA-LSTMs benefit more than XGBoost from larger basin sets.

These observations suggest that models generalize knowledge across basins. We illustrate this concept with the example of a basin that consistently exhibits low flows during its training period but has a high-flow event in the test period. If we train a model on just this one basin, the NSE on the test period will likely be poor, as the model has never seen high flows. When we train the model on a large number of basins, however, chances are that there exists a similar basin that *does* have high flows in its training period. With this information, a hypothetical model that transfers knowledge could produce good predictions despite its lack of suitable training data for one basin. A recent study of Kratzert et al. on the same dataset supports this conclusion, as it shows that data-driven models can

predict ungauged basins (i.e., basins that were not seen during training) more successfully than physically-based models [24]. The fact that the models continue to perform well in such a setup further endorses our interpretation of shared knowledge across space.

Chapter 7

Conclusion

In this thesis, we explored applications of machine learning to streamflow prediction. First, we presented the Cuizinart as an easy-to-use interactive web tool that alleviates the subsetting of large environmental datasets. Second, we demonstrated in two case studies how data-driven approaches can outperform existing, physically-based streamflow prediction models. Moreover, we designed a combined physically-based and data-driven model that yields more accurate predictions than models of either paradigm individually. The data-driven component corrects the physically-based component's predictions and provides a metric of the physically-based states' trustworthiness. Motivated by the results of the case study, we lastly evaluated the effect of limited training data on the quality of data-driven predictions. We estimated optimal input sequence lengths for tree- and LSTM-based streamflow prediction models and quantitatively highlighted the importance of training models not only on long training periods, but also on large sets of basins. This result indicates that data-driven models can transfer knowledge across basins of similar characteristics, which is especially promising towards obtaining more accurate predictions on ungauged basins.

As future work, we see great merit in combined physically-based and data-driven models under the paradigm of "theory-guided data science" [23]. For instance, algorithms such as Local Interpretable Model-Agnostic Explanations (LIME) could explain which hydrologic scenarios cause incorrect physically-based predictions and therefore entail large corrections [38]. Subsequently, hydrologists could improve the underlying physically-based model to better adapt to these scenarios. Closely related to this approach is the idea of "physics-guided machine learning", where we incorporate domain knowledge directly into data-driven architectures to ensure predictions that comply with known physical laws [19]. One example for such prior knowledge in streamflow prediction is the conservation of mass:

Assuming a closed system, the total amount of streamflow, evapotranspiration, and difference in stored water must be identical to the total precipitation across time. In line with these efforts towards tighter integration of physically-based and data-driven models, another promising research direction are spatially distributed data-driven models. Like the convolutional LSTM in our case study on Lake Erie, such models operate directly on the spatial grid of input variables and therefore more naturally generalize across space than lumped models. It is, however, challenging to train such models, because the high-dimensional input grid requires copious training sets. Nevertheless, distributed data-driven architectures could allow for better coupling of the two modeling paradigms, since state-of-the-art physically-based models are often distributed setups as well.

In summary, we believe that joint efforts of physically-based and data-driven modeling can help improve hydrologic models, redress the ostensible trade-off between accuracy and interpretability, and therefore ultimately advance hydrologic understanding.

References

[1] Nans Addor, Grey Nearing, Cristina Prieto, Andrew Newman, Nataliya Le Vine, and Martyn P. Clark. A ranking of hydrological signatures based on their predictability in space. *Water Resources Research*, 54(11):8792–8812, 2018.

[2] Nans Addor, Andrew Newman, Naoki Mizukami, and Martyn P. Clark. The CAMELS data set: catchment attributes and meteorology for large-sample studies. *Hydrology and Earth System Sciences*, 21(10):5293–5313, 2017.

[3] Zilefac Elvis Asong, Howard Wheater, John Pomeroy, Alain Pietroniro, and Mohamed Elshamy. A bias-corrected 3-hourly 0.125 gridded meteorological forcing data set (1979 – 2016) for land surface modeling in North America, 2018.

[4] Elena Carla Carcano, Paolo Bartolini, Marco Muselli, and Luigi Piroddi. Jordan recurrent neural network versus IHACRES in modelling daily streamflows. *Journal of Hydrology*, 362(3):291–307, 2008.

[5] Giuseppe Carleo, Ignacio Cirac, Kyle Cranmer, Laurent Daudet, Maria Schuld, Naftali Tishby, Leslie Vogt-Maranto, and Lenka Zdeborová. Machine learning and the physical sciences. *Reviews of Modern Physics*, 91:045002, 2019.

[6] F.-John Chang, Li-Chiu Chang, and Hau-Lung Huang. Real-time recurrent learning neural network for stream-flow forecasting. *Hydrological Processes*, 16(13):2577–2588, 2002.

[7] Tianqi Chen and Carlos Guestrin. XGBoost: A scalable tree boosting system. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '16, pages 785–794. ACM, 2016.

[8] Jacob Cohen. *Statistical power analysis for the behavioral sciences*. Routledge, 2013.

[9] Thomas E. Croley and Chansheng He. Great Lakes large basin runoff model. In *Proceedings, Second Federal Interagency Hydrologic Conference*, 2002.

[10] Myron B. Fiering. *Streamflow synthesis*, volume 1. Harvard University Press, 1967.

[11] Jacob Gardner, Geoff Pleiss, Kilian Q. Weinberger, David Bindel, and Andrew G. Wilson. GPyTorch: Blackbox matrix-matrix gaussian process inference with GPU acceleration. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, editors, *Advances in Neural Information Processing Systems 31*, pages 7576–7586. Curran Associates, Inc., 2018.

[12] Martin Gauch, Juliane Mai, and Jimmy Lin. The proper care and feeding of CAMELS: How limited training data affects streamflow prediction, 2019. *arXiv:1911.07249*.

[13] Hoshin V. Gupta, Harald Kling, Koray K. Yilmaz, and Guillermo F. Martinez. Decomposition of the mean squared error and NSE performance criteria: implications for improving hydrological modelling. *Journal of Hydrology*, 377(1-2):80–91, 2009.

[14] Albert H. Halff, Henry M. Halff, and Masoud Azmoodeh. Predicting runoff from rainfall using neural networks. In *Proceedings of the Symposium Sponsored by the Hydraulics Division of ASCE*, pages 760–765, 1993.

[15] Joseph J. Hamman, Bart Nijssen, Theodore J. Bohn, Diana R. Gergel, and Yixin Mao. The Variable Infiltration Capacity model version 5 (VIC-5): infrastructure improvements for new applications and reproducibility. *Geoscientific Model Development*, 11(8), 2018.

[16] Tony Hey, Stewart Tansley, and Kristin M. Tolle, editors. *The Fourth Paradigm: Data-Intensive Scientific Discovery*. Microsoft Research, 2009.

[17] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural Computation*, 9(8):1735–1780, 1997.

[18] M. Hrachowitz, H.H.G. Savenije, G. Blöschl, J.J. McDonnell, M. Sivapalan, J.W. Pomeroy, B. Arheimer, T. Blume, M.P. Clark, U. Ehret, F. Fenicia, J.E. Freer, A. Gelfan, H.V. Gupta, D.A. Hughes, R.W. Hut, A. Montanari, S. Pande, D. Tetzlaff, P.A. Troch, S. Uhlenbrook, T. Wagener, H.C. Winsemius, R.A. Woods, E. Zehe, and C. Cudennec. A decade of predictions in ungauged basins (PUB)—a review. *Hydrological Sciences Journal*, 58(6):1198–1255, 2013.

[19] Xiaowei Jia, Jared Willard, Anuj Karpatne, Jordan S. Read, Jacob A. Zwart, Michael Steinbach, and Vipin Kumar. Physics-guided machine learning for scientific discovery: An application in simulating lake temperature profiles, 2020. *arXiv:2001.11086*.

[20] Kaggle Team. Rossmann store sales, winner's interview: 1st place, Gert Jacobusse. No Free Hunch, Dec 2015. http://blog.kaggle.com/2015/12/21/rossmann-store-sales-winners-interview-1st-place-gert/, accessed Nov 2019.

[21] Kaggle Team. Grupo Bimbo inventory demand, winners' interview: Clustifier & Alex & Andrey. Kaggle Blog, Sep 2016. https://medium.com/kaggle-blog/grupo-bimbo-inventory-demand-winners-interview-clustifier-alex-andrey-1e3b6cec8a20, accessed Feb 2020.

[22] Kaggle Team. Rossmann store sales, winner's interview: 2nd place, Nima Shahbazi. No Free Hunch, Mar 2016. http://blog.kaggle.com/2016/02/03/rossmann-store-sales-winners-interview-2nd-place-nima-shahbazi/, accessed Nov 2019.

[23] Anuj Karpatne, Gowtham Atluri, James H. Faghmous, Michael Steinbach, Arindam Banerjee, Auroop Ganguly, Shashi Shekhar, Nagiza Samatova, and Vipin Kumar. Theory-guided data science: A new paradigm for scientific discovery from data. *IEEE Transactions on Knowledge and Data Engineering*, 29(10):2318–2331, 2017.

[24] Frederik Kratzert, Daniel Klotz, Mathew Herrnegger, Alden K. Sampson, Sepp Hochreiter, and Grey Nearing. Toward improved predictions in ungauged basins: Exploiting the power of machine learning. *Water Resources Research*, 2019.

[25] Frederik Kratzert, Daniel Klotz, Guy Shalev, Günter Klambauer, Sepp Hochreiter, and Grey Nearing. Benchmarking a catchment-aware Long Short-Term Memory network (LSTM) for large-scale hydrological modeling. *Hydrology and Earth System Sciences Discussions*, pages 1–32, 2019.

[26] Xu Liang, Dennis P. Lettenmaier, Eric F. Wood, and Stephen J. Burges. A simple hydrologically based model of land surface water and energy fluxes for general circulation models. *Journal of Geophysical Research*, 99(D7):14415–14428, 1994.

[27] Yue Liu, Tianlu Zhao, Wangwei Ju, and Siqi Shi. Materials discovery and design using machine learning. *Journal of Materiomics*, 3(3):159–177, 2017. High-throughput Experimental and Modeling Research toward Advanced Batteries.

[28] Zhipeng Luo, Jianqiang Huang, Ke Hu, Xue Li, and Peng Zhang. AccuAir: Winning solution to air quality prediction for KDD Cup 2018. In *Proceedings of the 25th ACM*

SIGKDD International Conference on Knowledge Discovery & Data Mining, KDD '19, pages 1842–1850. ACM, 2019.

[29] Juliane Mai, Bryan Tolson, Hongren Shen, Étienne Gaborit, Vincent Fortin, Milena Dimitrijevic, Nicolas Gasset, Dorothy Durnford, Young Lan Shin, Trica A. Stadnyk, Lauren M. Fry, Tim Hunter, Andrew Gronewold, Joseph Smith, Lacey Mason, Laura Read, Katelyn FitzGerald, Kevin Sampson, Alan F. Hamlet, Frank Seglenieks, Shervan Gharari, Saman Razavi, Amin Haghnegahdar, Daniel G. Princz, and Alain Pietroniro. The Great Lakes Runoff Inter-comparison Project for Lake Erie (GRIP-E). *AGU Fall Meeting Abstracts*, 2018.

[30] Juliane Mai, Bryan Tolson, Hongren Shen, Étienne Gaborit, Vincent Fortin, Milena Dimitrijevic, Nicolas Gasset, Dorothy Durnford, Young Lan Shin, Tricia A. Stadnyk, Hervé R. Awoye, Lauren M. Fry, Emily A. Bradley, Tim Hunter, Andrew Gronewold, Joseph Smith, Lacey Mason, Laura Read, Katelyn FitzGerald, Kevin Sampson, Alan F. Hamlet, Frank Seglenieks, André G. T. Temgoua, Shervan Gharari, Saman Razavi, Amin Haghnegahdar, Mohamed Elshamy, Daniel G. Princz, Alain Pietroniro, Xiaojing Ni, Yongping Yuan, Mohammad R. Najafi, Melika Rahimimovaghar, Martin Gauch, Jimmy Lin, and Raphael Tang. The runoff model-intercomparison project over Lake Erie and the Great Lakes. *AGU Fall Meeting Abstracts*, 2019.

[31] Edwin P. Maurer, Andrew W. Wood, Jennifer C. Adam, Dennis P. Lettenmaier, and Bart Nijssen. A long-term hydrologically based dataset of land surface fluxes and states for the conterminous United States. *Journal of Climate*, 15(22):3237–3251, 2002.

[32] Ron C. Mittelhammer, George G. Judge, and Douglas J. Miller. *Econometric Foundations*, pages 73–74. Cambridge University Press, 2000.

[33] Naoki Mizukami, Oldrich Rakovec, Andrew Newman, Martyn P. Clark, Andrew W. Wood, Hoshin V. Gupta, and Rohini Kumar. On the choice of calibration metrics for "high-flow" estimation using hydrologic models. *Hydrology and Earth System Sciences*, 23(6):2601–2614, 2019.

[34] Grey Nearing, Frederik Kratzert, Alden K. Sampson, Craig S. Pelissier, Daniel Klotz, Jonathan Frame, and Hoshin Gupta. What role does hydrological science play in the age of machine learning?, 2020. EarthArXiv.

[35] Andrew Newman, Naoki Mizukami, Martyn P. Clark, Andrew W. Wood, Bart Nijssen, and Grey Nearing. Benchmarking of a physically based hydrologic model. *Journal of Hydrometeorology*, 18(8):2215–2225, 2017.

[36] Andrew Newman, Kevin Sampson, Martyn P. Clark, Andy Bock, Roland Viger, and David Blodgett. A large-sample watershed-scale hydrometeorological dataset for the contiguous USA, 2014.

[37] Christopher Olah. Understanding LSTM networks. colah's blog, Aug 2015. https://colah.github.io/posts/2015-08-Understanding-LSTMs/, accessed Feb 2020.

[38] Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. "Why should I trust you?": Explaining the predictions of any classifier. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '16, pages 1135–1144. ACM, 2016.

[39] Luis Samaniego, Rohini Kumar, and Sabine Attinger. Multiscale parameter regionalization of a grid-based hydrologic model at the mesoscale. *Water Resources Research*, 46(5), 2010.

[40] Shlomo S. Sawilowsky. New effect size rules of thumb. *Journal of Modern Applied Statistical Methods*, 8(2):597–599, 2009.

[41] Xingjian Shi, Zhourong Chen, Hao Wang, Dit-Yan Yeung, Wai-Kin Wong, and Wangchun Woo. Convolutional LSTM network: A machine learning approach for precipitation nowcasting. In *Advances in Neural Information Processing Systems 28*, pages 802–810. Curran Associates, Inc., 2015.

APPENDICES

A Training Procedures

We train all our models using Python 3.7.3, PyTorch 1.1.0, and CUDA 9.0. For CPU-intensive training such as XGBoost, we use Intel Xeon E5-2683 and E5-2699 v4 CPUs; for GPU-intensive computations, we use NVIDIA P100 Pascal and V100 Volta GPUs.

A.1 LSTM-Based Models

We train our LSTM-based models for 30 epochs with an initial learning rate of 0.001 that reduces to 0.0005 after ten epochs and to 0.0001 after another ten epochs. We feed batches of 256 samples into the networks, which consist of one 256-neuron hidden layer with a dropout rate of 0.4. For the EA-LSTM model, we use the open-source implementation of Kratzert et al. (Git version `2dd199e`, https://github.com/kratzert/ealstm_regional_modeling).

A.2 XGBoost Models

For XGBoost (Git version `96cd7ec`, https://github.com/dmlc/xgboost), we find suitable hyperparameters in two three-fold cross-validated random searches. First, we search for good tree parameters (maximum tree depth, minimum child weight, column sampling, gamma) in 5000 random samples. Next, we use the found parameters in a 100-iteration random search to find regularization parameters (alpha, lambda). Both random searches fit up to 100 trees at a learning rate of 0.25 and stop after 50 rounds without improvement.

In our study on the effect of limited training data on the CAMELS dataset, we find that longer sequence lengths work better for larger datasets. Hence, we perform the hyperparameter search for sequence length 10 on three years and 13 basins, for sequence length

30 on six training years and 53 basins, and for sequence length 100 on nine years and 265 basins. Table A.1 lists the final hyperparameters for each sequence length in this study. After parameter tuning, we train the XGBoost models at a learning rate of 0.08 for up to 20 000 iterations; however, we stop once the NSE′-loss on a validation set of 10 % of the training data does not improve for 100 rounds.

Table A.1: Final XGBoost hyperparameters for each input sequence length in our study on limited data with the CAMELS dataset.

	Sequence length		
Parameter	10	30	100
`n_estimators`	20 000	20 000	20 000
`early_stopping_rounds`	100	100	100
`learning_rate`	0.08	0.08	0.08
`max_depth`	4	6	7
`min_child_weight`	1	1	9
`colsample_bytree`	0.962	0.400	0.884
`colsample_bylevel`	0.916	0.968	0.485
`gamma`	1.293	1.005	4.586
`reg_alpha`	1.091	18.944	24.190
`reg_lambda`	2.738	3.704	67.595
`subsample`	0.9	0.9	0.9

B Attributes in the CAMELS Dataset

Table B.1 shows the 27 static basin attributes from the CAMELS dataset we use in our study on the effect of limited training data (cf. Chapter 6).

Table B.1: CAMELS static basin attributes used in this study. (Source: Addor et al. [2])

Variable name	Explanation
p_mean	Mean daily precipitation.
pet_mean	Mean daily potential evapotranspiration.
aridity	Ratio of mean PET to mean precipitation.
p_seasonality	Seasonality and timing of precipitation. Estimated by representing annual precipitation and temperature as sin waves. Positive (negative) values indicate precipitation peaks during the summer (winter). Values of approx. 0 indicate uniform precipitation throughout the year.
frac_snow_daily	Fraction of precip. falling on days with temperatures $< 0\,°C$.
high_prec_freq	Frequency of high precipitation days ($\leq 5 \times$ p_mean).
high_prec_dur	Average duration of high precipitation events (number of consecutive days with $\leq 5 \times$ p_mean).
low_prec_freq	Frequency of dry days (< 1 mm/day).
low_prec_dur	Average duration of dry periods (number of consecutive days with precipitation < 1 mm/day).
elev_mean	Catchment mean elevation.
slope_mean	Catchment mean slope.
area_gages2	Catchment area.
forest_frac	Forest fraction.
lai_max	Maximum monthly mean of leaf area index (LAI).
lai_diff	Difference between the max. and min. mean of the LAI.
gvf_max	Maximum monthly mean of green vegetation fraction (GVF).
gvf_diff	Difference between the max. and min. monthly mean GVF.
soil_depth_pelletier	Depth to bedrock (maximum 50 m).
soil_depth_statsgo	Soil depth (maximum 1.5 m).
soil_porosity	Volumetric porosity.
soil_conductivity	Saturated hydraulic conductivity.
max_water_content	Maximum water content of the soil.
sand_frac	Fraction of sand in the soil.
silt_frac	Fraction of silt in the soil.
clay_frac	Fraction of clay in the soil.
carb_rocks_frac	Fraction of the catchment area characterized as "Carbonate sedimentary rocks".
geol_permeability	Surface permeability (log10).